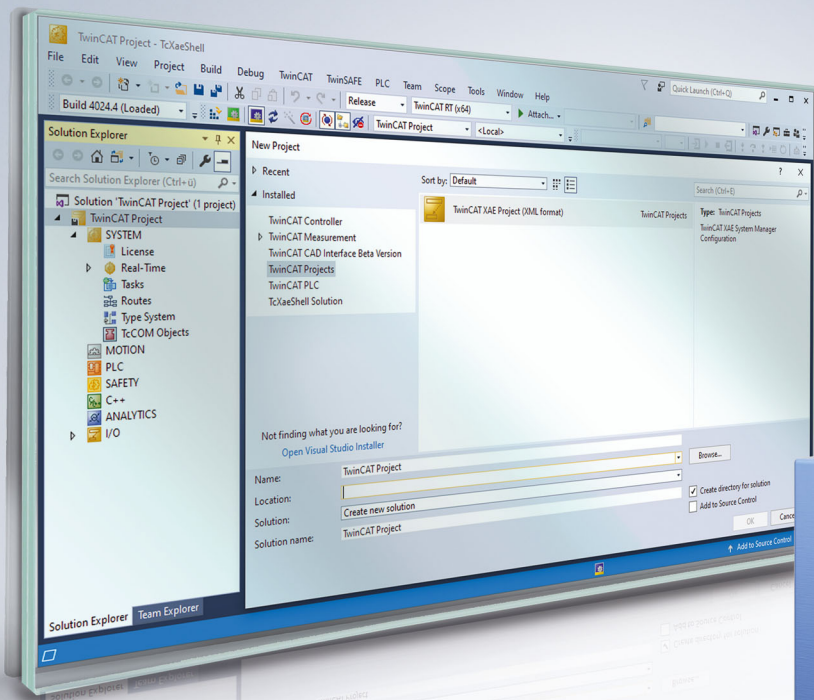


# BECKHOFF New Automation Technology

Manual | EN

# TF3520

TwinCAT 3 | Analytics Storage Provider





# Table of contents

<b>1</b>	<b>Foreword</b> .....	<b>5</b>
1.1	Notes on the documentation .....	5
1.2	For your safety .....	5
1.3	Notes on information security.....	7
<b>2</b>	<b>Overview</b> .....	<b>8</b>
<b>3</b>	<b>Installation</b> .....	<b>9</b>
3.1	System requirements .....	9
3.2	Installation .....	9
3.3	Licensing .....	12
3.4	Installing the TwinCAT/BSD .....	14
<b>4</b>	<b>Analytics Workflow - First Steps</b> .....	<b>16</b>
4.1	Recording data from the machine .....	16
4.2	Communication .....	19
4.3	Historicize data.....	20
4.4	Import Analytics Files .....	27
4.5	Analyse data .....	28
4.6	24h Analytics application.....	33
<b>5</b>	<b>Technical introduction</b> .....	<b>42</b>
<b>6</b>	<b>Configuration</b> .....	<b>44</b>
6.1	Service .....	44
6.2	Databases/Stores.....	46
6.2.1	Analytics Binary File.....	46
6.2.2	Microsoft SQL .....	47
6.2.3	Microsoft Azure Blob .....	49
6.3	Recorder .....	51
6.4	Working with Historical Data .....	55
6.5	Console Configurator/Client .....	59
6.5.1	Configurator .....	60
6.5.2	Client.....	61
6.5.3	Batch files for control.....	64
<b>7</b>	<b>PLC API</b> .....	<b>68</b>
7.1	Function blocks .....	68
7.1.1	Topic Architecture .....	68
7.1.2	FB_ALY_StorageProvider.....	81
7.2	Data types .....	87
7.2.1	ST_ConnectionSettings .....	87
7.2.2	ST_ALY_SP_Config.....	87
7.2.3	E_CancelType.....	88
7.2.4	E_RawDataFormat.....	88
7.2.5	E_RecordMode .....	88
7.2.6	E_ReloadType .....	89
7.2.7	E_RingBufferMode.....	89
7.2.8	E_SetGetHistoricalDataState.....	89

---

7.2.9	E_SymbolMode.....	90
<b>8</b>	<b>Samples .....</b>	<b>91</b>
8.1	PLC Client .....	91
<b>9</b>	<b>Appendix .....</b>	<b>96</b>
9.1	FAQ - frequently asked questions and answers .....	96
9.2	Third-party components .....	96

# 1 Foreword

## 1.1 Notes on the documentation

This description is intended exclusively for trained specialists in control and automation technology who are familiar with the applicable national standards.

For installation and commissioning of the components, it is absolutely necessary to observe the documentation and the following notes and explanations.

The qualified personnel is obliged to always use the currently valid documentation.

The responsible staff must ensure that the application or use of the products described satisfies all 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 notice.

No claims to modify 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 and licensed trademarks of Beckhoff Automation GmbH.

If third parties make use of designations or trademarks used in this publication for their own purposes, this could infringe upon the rights of the owners of the said designations.

### Patents

The EtherCAT Technology is covered, including but not limited to the following patent applications and patents:

EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702  
and similar applications and registrations in several other countries.

## EtherCAT®

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

### Copyright

© Beckhoff Automation GmbH & Co. KG, Germany.

The distribution and reproduction of this document as well as the use and communication of its contents without express authorization are prohibited.

Offenders will be held liable for the payment of damages. All rights reserved in the event that a patent, utility model, or design are registered.

## 1.2 For your safety

### Safety regulations

Read the following explanations for your safety.

Always observe and follow product-specific safety instructions, which you may find at the appropriate places in this document.

**Exclusion of liability**

All the components are supplied in particular hardware and software configurations which are 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 technology who are familiar with the applicable national standards.

**Signal words**

The signal words used in the documentation are classified below. In order to prevent injury and damage to persons and property, read and follow the safety and warning notices.

**Personal injury warnings****⚠ DANGER**

Hazard with high risk of death or serious injury.

**⚠ WARNING**

Hazard with medium risk of death or serious injury.

**⚠ CAUTION**

There is a low-risk hazard that could result in medium or minor injury.

**Warning of damage to property or environment****NOTICE**

The environment, equipment, or data may be damaged.

**Information on handling the product**

This information includes, for example:  
recommendations for action, assistance or further information on the product.

## 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

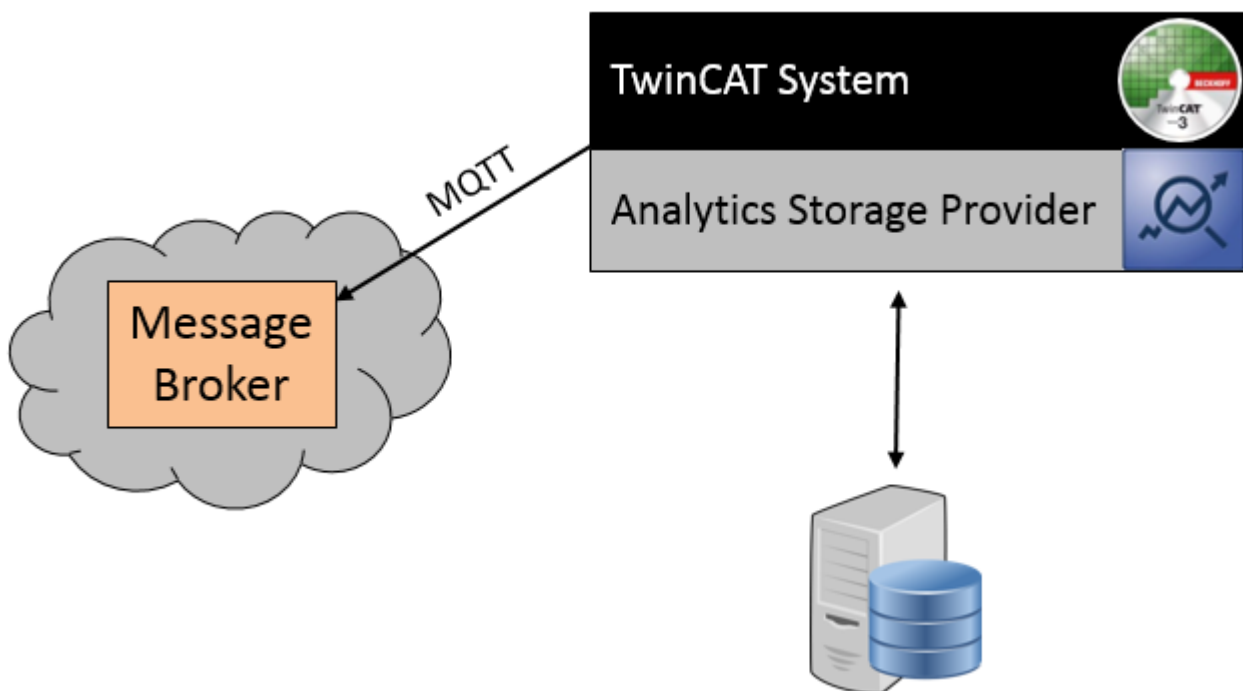
With the TwinCAT Analytics Storage Provider, Beckhoff offers a way to continuously store high-resolution data in a binary format. The decisive factor is that the user does not have to worry about data storage. The storage provider takes care of this automatically. The configuration is done with a few clicks in engineering. Complex SQL commands are not necessary. Classic databases can be used, but also binary blob stores.

### Components

- TwinCAT Analytics Storage Provider service: A Windows service that manages the communication.
- TwinCAT Analytics Storage Provider PLC Library: A TwinCAT 3 PLC library with functions for controlling the storage provider from a PLC application.

### Principle of operation

The Analytics Storage Provider receives and sends data via MQTT communication protocol. For this purpose, it is connected to a native MQTT message broker in the network and on the other side to the corresponding data sink.



### Supported databases/storage

- 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.

Technical data Service	TF3520 TC 3 Analytics Storage Provider
Target system	Windows 7, Windows 8, Windows 10
.NET Framework	.Net 4.5.1 or higher
Min. TwinCAT version	3.1.4022.25
Min. TwinCAT level	TC1000 TC3   ADS

Technical data Library	TF3520 TC 3 Analytics Storage Provider
Target system	Windows 7, Windows 8, Windows 10
Min. TwinCAT version	3.1.4022.29
Min. TwinCAT level	TC1200 TC3   PLC

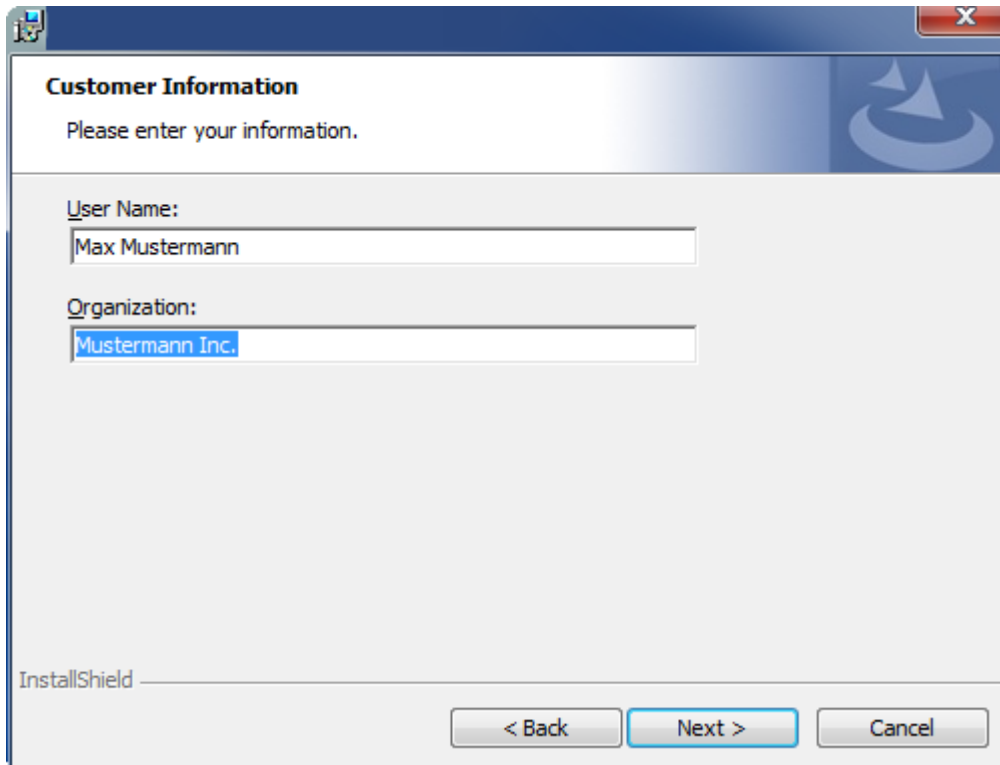
#### 3.2 Installation

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.



**Customer Information**

Please enter your information.

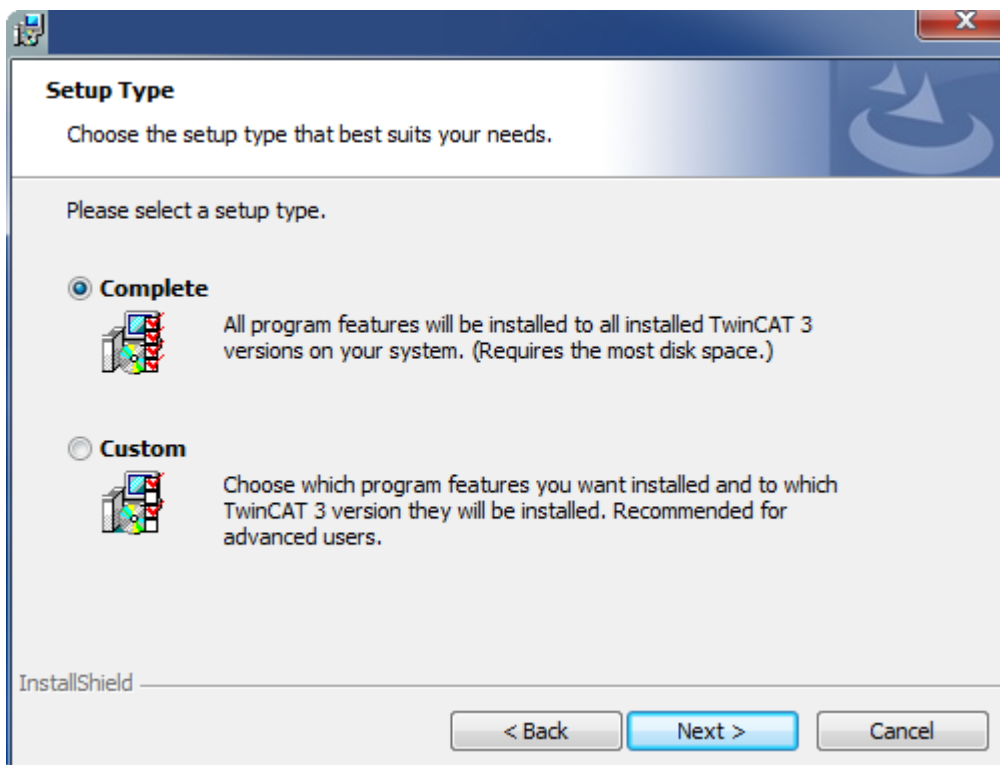
User Name:  
Max Mustermann

Organization:  
Mustermann Inc.

InstallShield

< Back   Next >   Cancel

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**.



**Setup Type**

Choose the setup type that best suits your needs.

Please select a setup type.

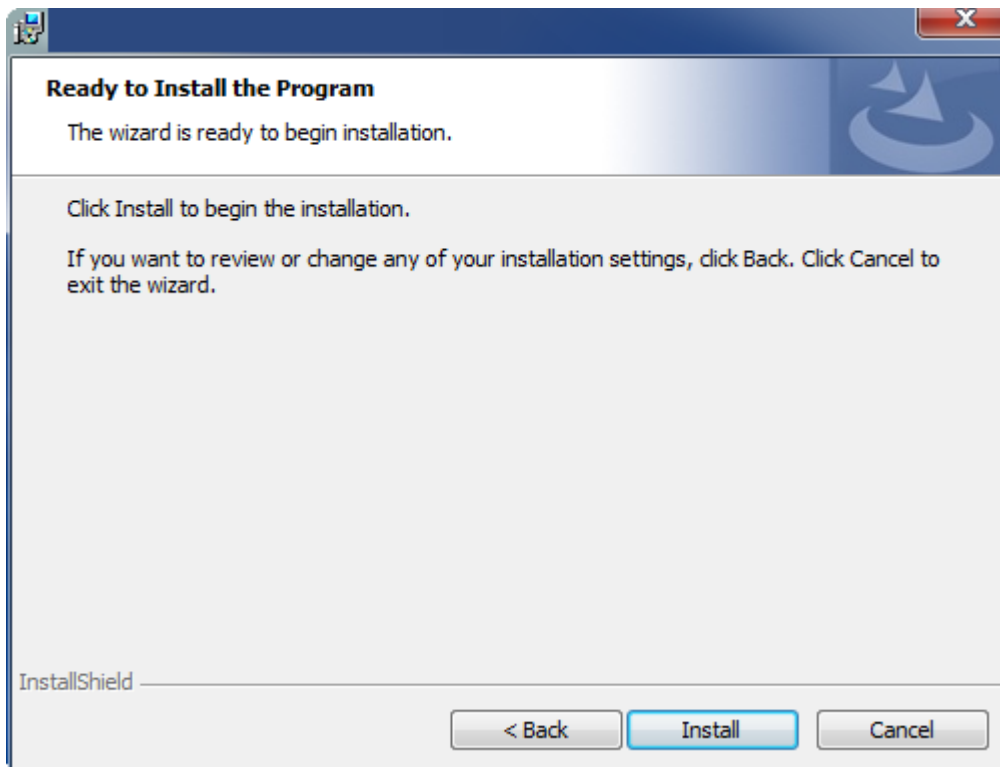
**Complete**  
All program features will be installed to all installed TwinCAT 3 versions on your system. (Requires the most disk space.)

**Custom**  
Choose which program features you want installed and to which TwinCAT 3 version they will be installed. Recommended for advanced users.

InstallShield

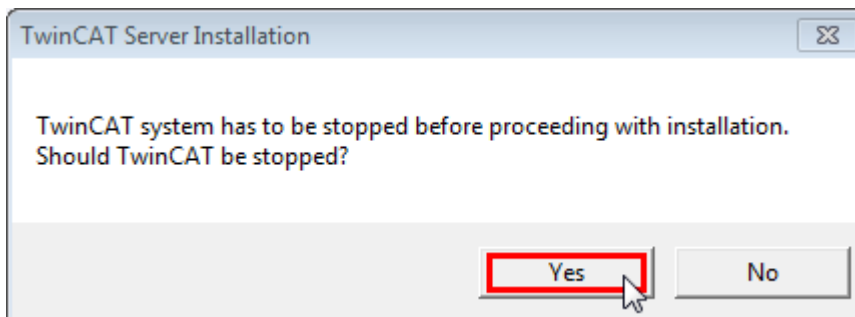
< Back   Next >   Cancel

5. Select **Next**, then **Install** to start the installation.

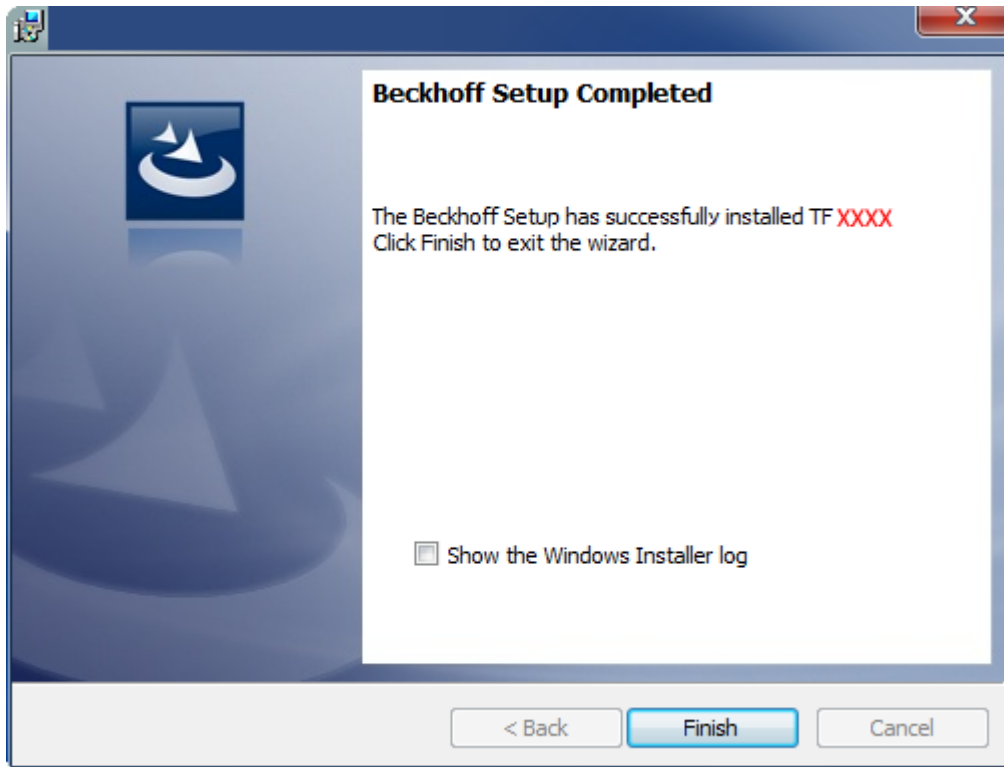


⇒ A dialog box informs you that the TwinCAT system must be stopped to proceed with the installation.

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](#)".

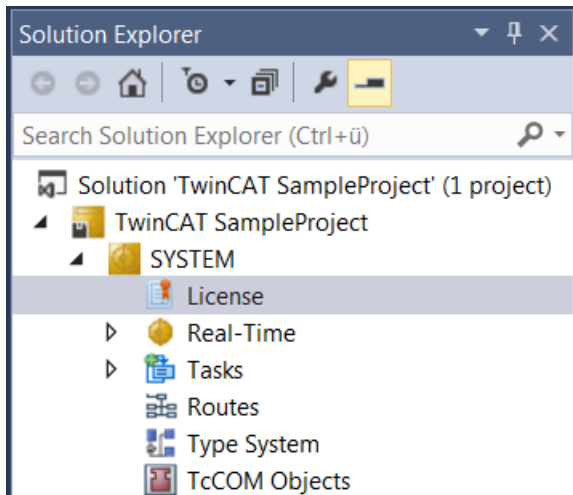
#### Licensing the 7-day test version of a TwinCAT 3 Function



A 7-day test version cannot be enabled for a [TwinCAT 3 license dongle](#).

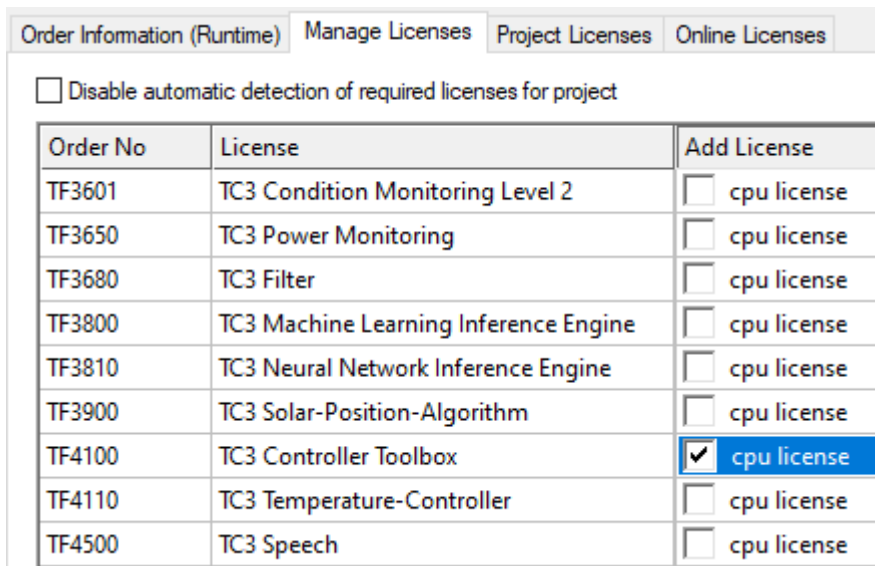
1. Start the TwinCAT 3 development environment (XAE).
2. Open an existing TwinCAT 3 project or create a new project.
3. If you want to activate the license for a remote device, set the desired target system. To do this, select the target system from the **Choose Target System** drop-down list in the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

- In the **Solution Explorer**, double-click **License** in the **SYSTEM** subtree.



⇒ The TwinCAT 3 license manager opens.

- 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").



- Open the **Order Information (Runtime)** tab.
  - ⇒ In the tabular overview of licenses, the previously selected license is displayed with the status "missing".

7. Click **7-Day Trial License...** to activate the 7-day trial license.

The screenshot shows a software window with tabs for 'Order Information (Runtime)', 'Manage Licenses', 'Project Licenses', and 'Online Licenses'. The 'License Device' section has a dropdown menu set to 'Target (Hardware Id)' and an 'Add...' button. Below it, 'System Id' is '2DB25408-B4CD-81DF-5488-6A3D9B49EF19' and 'Platform' is 'other (91)'. The 'License Request' section has a 'Provider' dropdown set to 'Beckhoff Automation' and a 'Generate File...' button. Below that are fields for 'License Id', 'Customer Id', and 'Comment'. The 'License Activation' section at the bottom has two buttons: '7 Days Trial License...' (highlighted with a red box) and 'License Response File...'.

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

The dialog box is titled 'Enter Security Code' and has a close button (X). It contains the text 'Please type the following 5 characters:'. Below this, the code 'Kg8T4' is displayed in a text field. To the right of the text field is an 'OK' button (highlighted with a red box) and a 'Cancel' button. Below the text field is another text field with a blue border and a red box around it, which is currently empty.

8. Enter the code exactly as it is displayed and confirm the entry.  
 9. Confirm the subsequent dialog, which indicates the successful activation.  
 ⇒ 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.

### 3.4 Installing the TwinCAT/BSD

The TwinCAT 3 Analytics Storage Provider Server is available as a package for TwinCAT/BSD in the package repository. Under the package name "TF3520 Analytics Storage Provider" it can be installed via the following command:

```
doas pkg install TF3520-Analytics-Storage-Provider
```

Further information about the [Package Server](#) can be found in the Embedded PC section of the TwinCAT/BSD manual.

After a system restart or restart of TwinCAT, the TwinCAT 3 Analytics Storage Provider Server is also started and can be configured by a client via MQTT.

#### **i** MQTT port enabling

To use the Analytics Storage Provider and [Console Configurator/Client](#) [▶ 59] under TwinCAT/BSD, the corresponding MQTT port must be enabled for communication. For more info see: [Port enabling under TwinCAT/BSD](#)

After installation, the Client.dll for the console is located under the path `/usr/local/etc/TwinCAT/Functions/TF3520-Analytics-Storage-Provider/Client`.

The Analytics Storage Provider service can be started by the following command if the license is activated:

```
doas service TcAnalyticsStorageProvider start
```

## 4 Analytics Workflow - First Steps

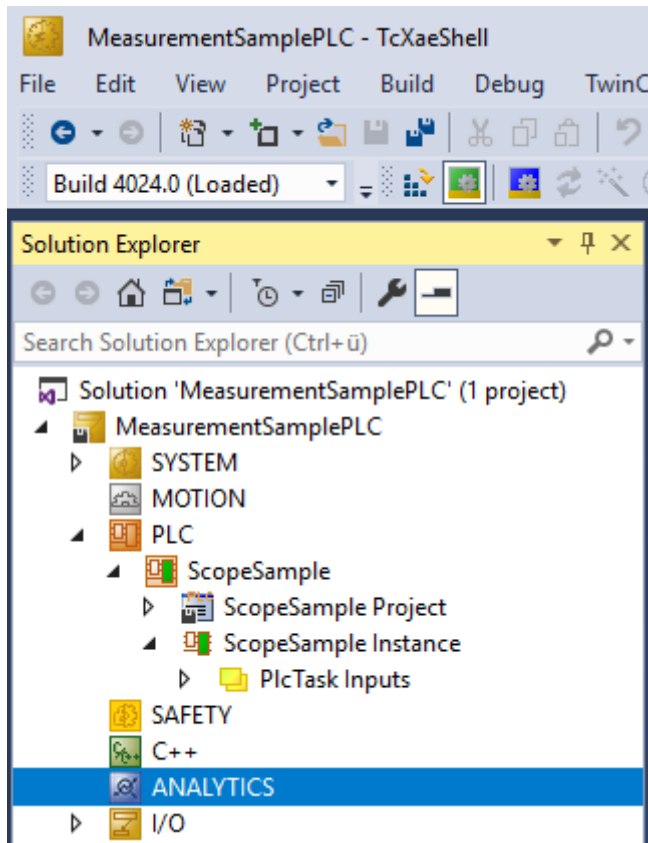
This step by step documentation presents the complete TwinCAT Analytics workflow. From the data acquisition over the communication and historizing up to the evaluation and analysis of the data and to the presentation of the data in web-based dashboard.

### 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.

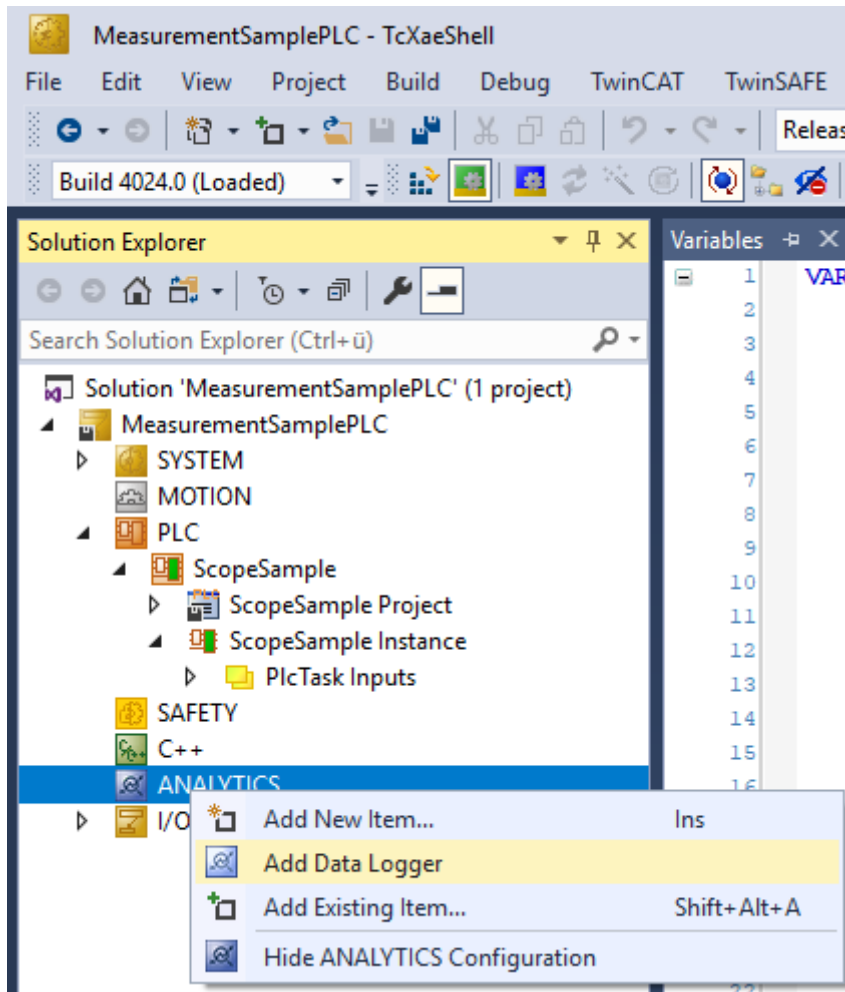
The TwinCAT Analytics Logger is installed with TwinCAT XAE and XAR. The Logger can act as MQTT Client to communicate the recorded data to a native MQTT Message Broker or store the data in the same data format in a local binary file. By the usage as MQTT Client the Logger is able to bypass short disconnects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

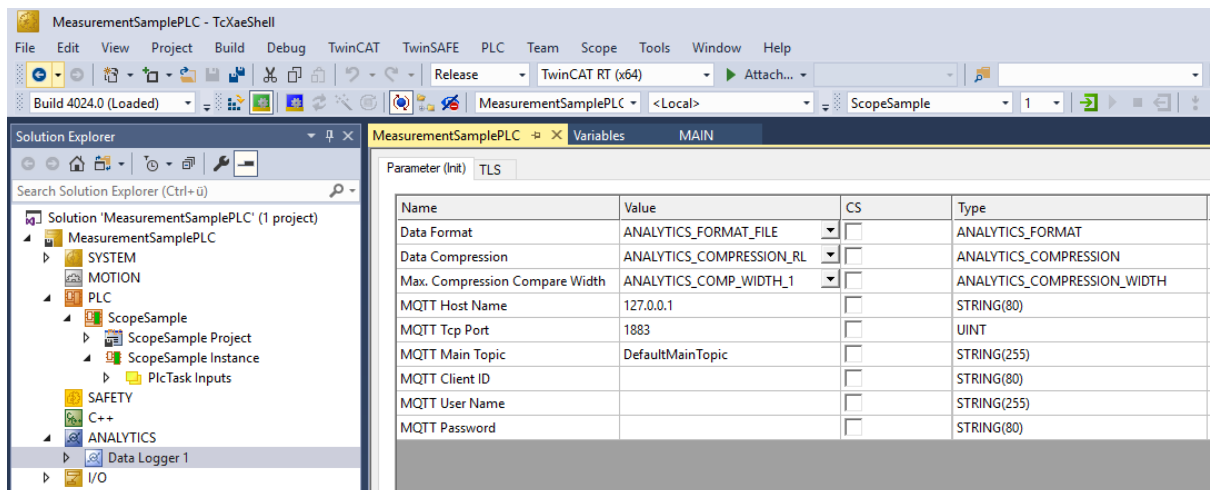




- Right click on this node and click on “Add Data Logger” to add one new instance to your configuration



- For configuring the base settings, please double click on the new tree item



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.

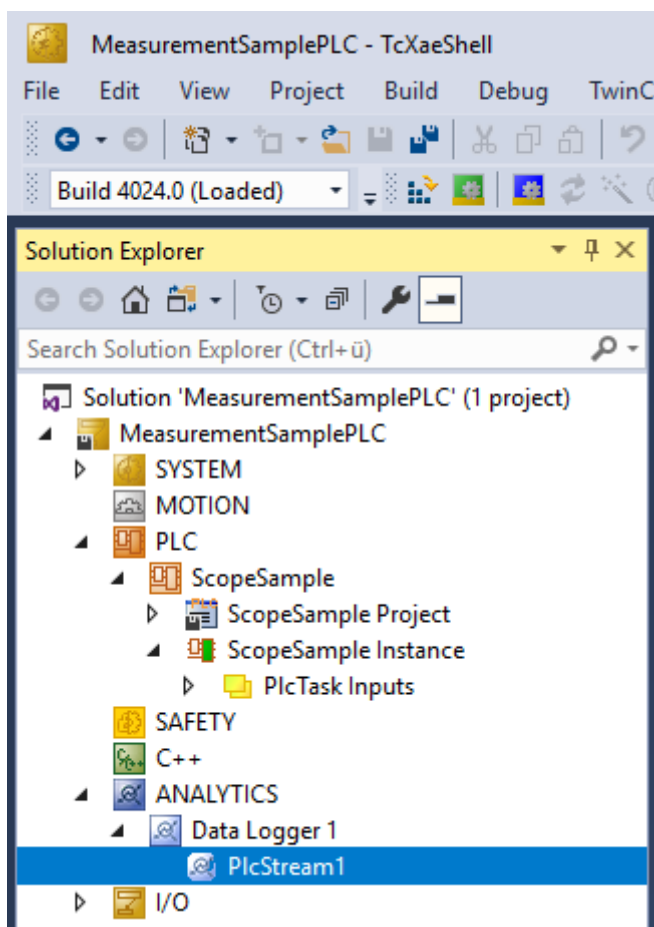
-BINARY: Data will be sent to the configured MQTT Message Broker. You can have multiple Logger in one TwinCAT project to communicate data to different MQTT Message Broker.

-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

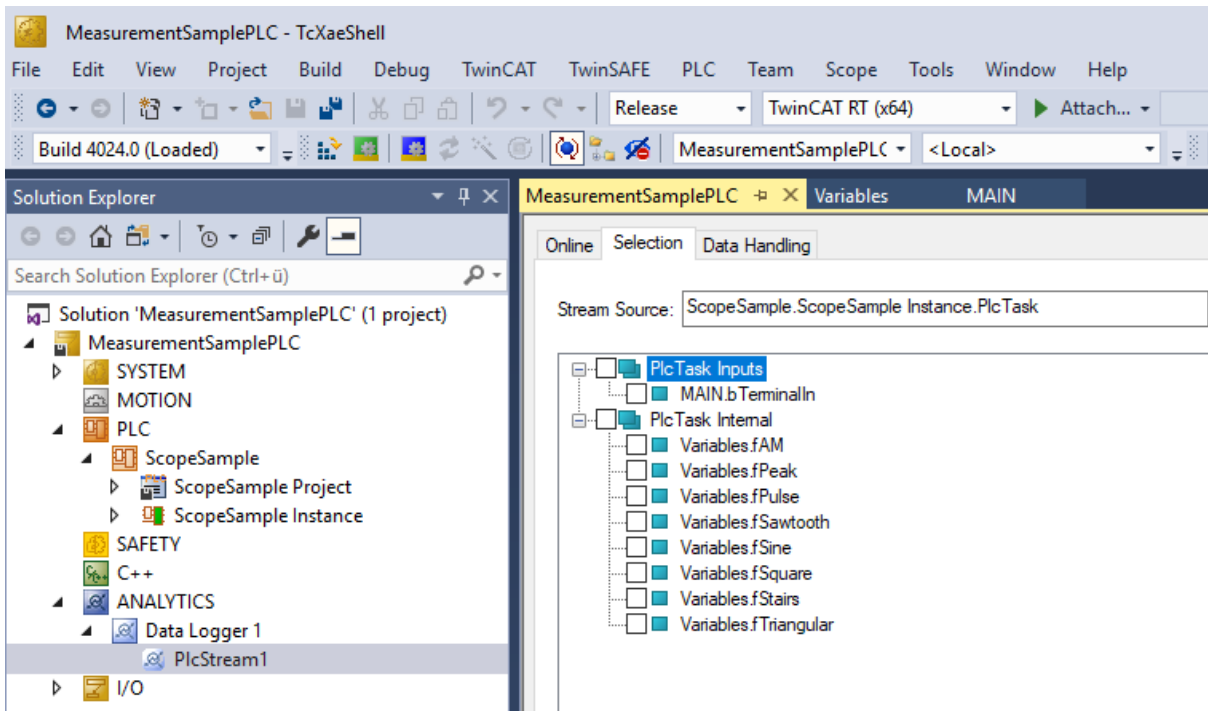
-At the TLS (Transport Layer Security) tab, security settings can be configured. TLS is a secure communication channel between client and server. By the usage of certificates, the TCP port 8883 is exclusively reserved for MQTT over TLS. Analytics Logger is supporting the modes CA Certificates, CA Certificates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

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

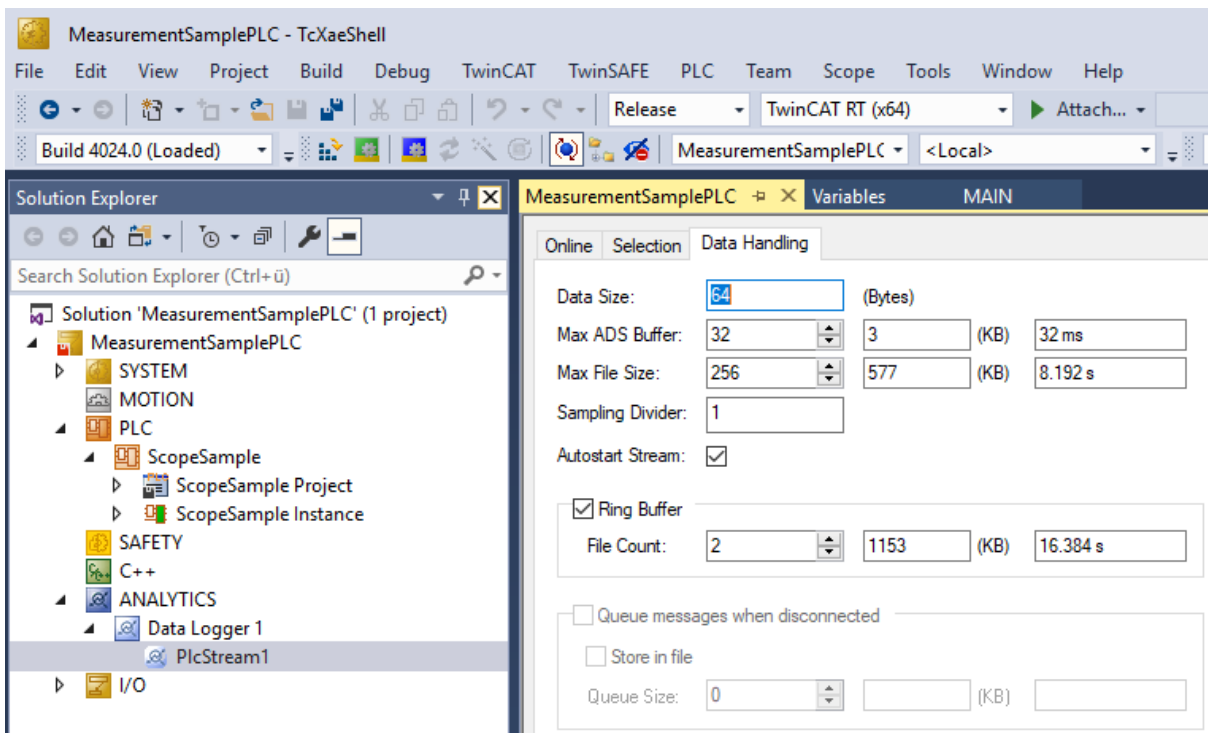


An additional device stream will be shown if your configuration provides an EtherCAT Process Image.

- In the stream a Selection tab is available to choose the variables that should be recorded

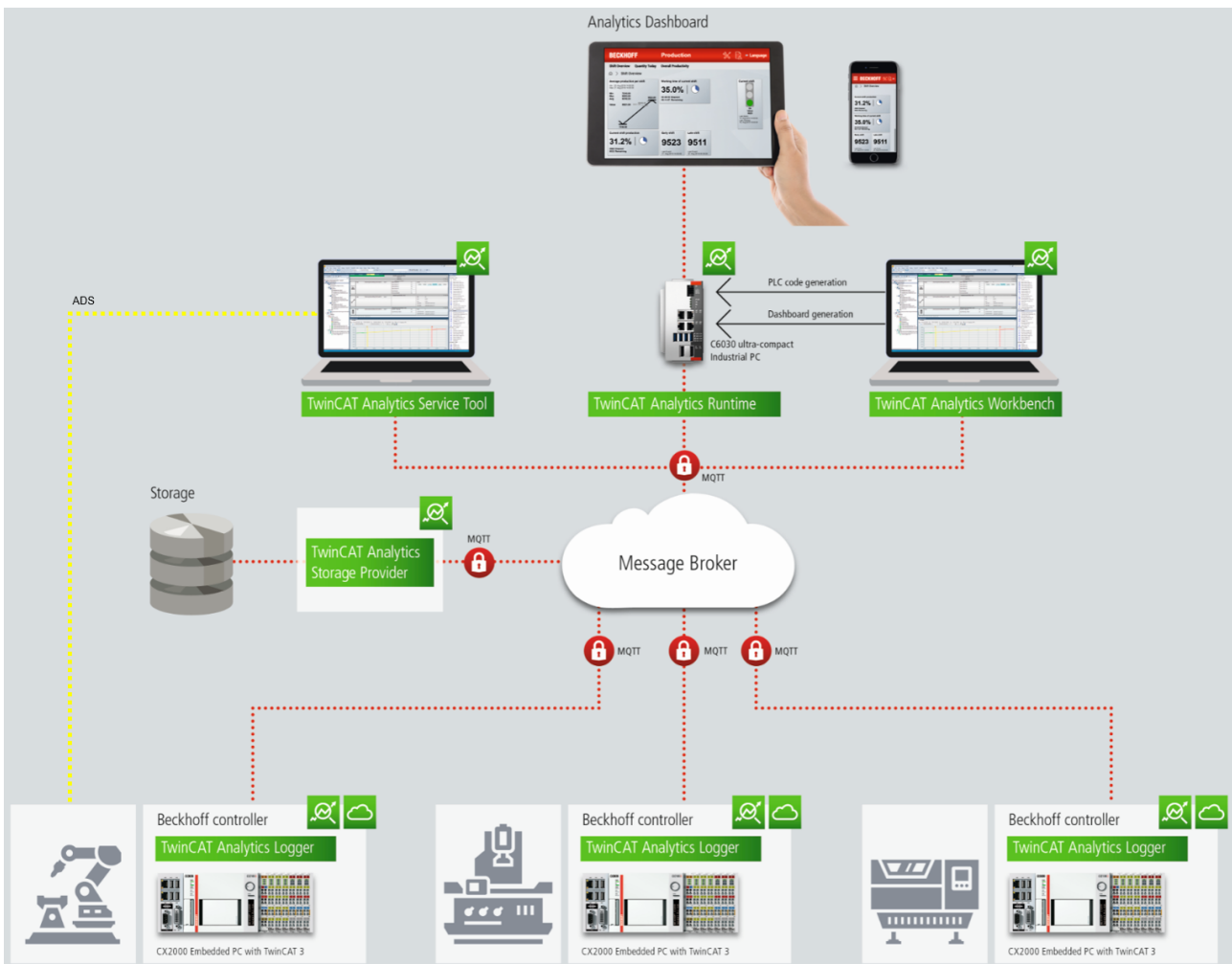


- Finally it is possible to change the package size for the frames or to configure the ring buffer for disconnects and file in the Data Handling tab.



## 4.2 Communication

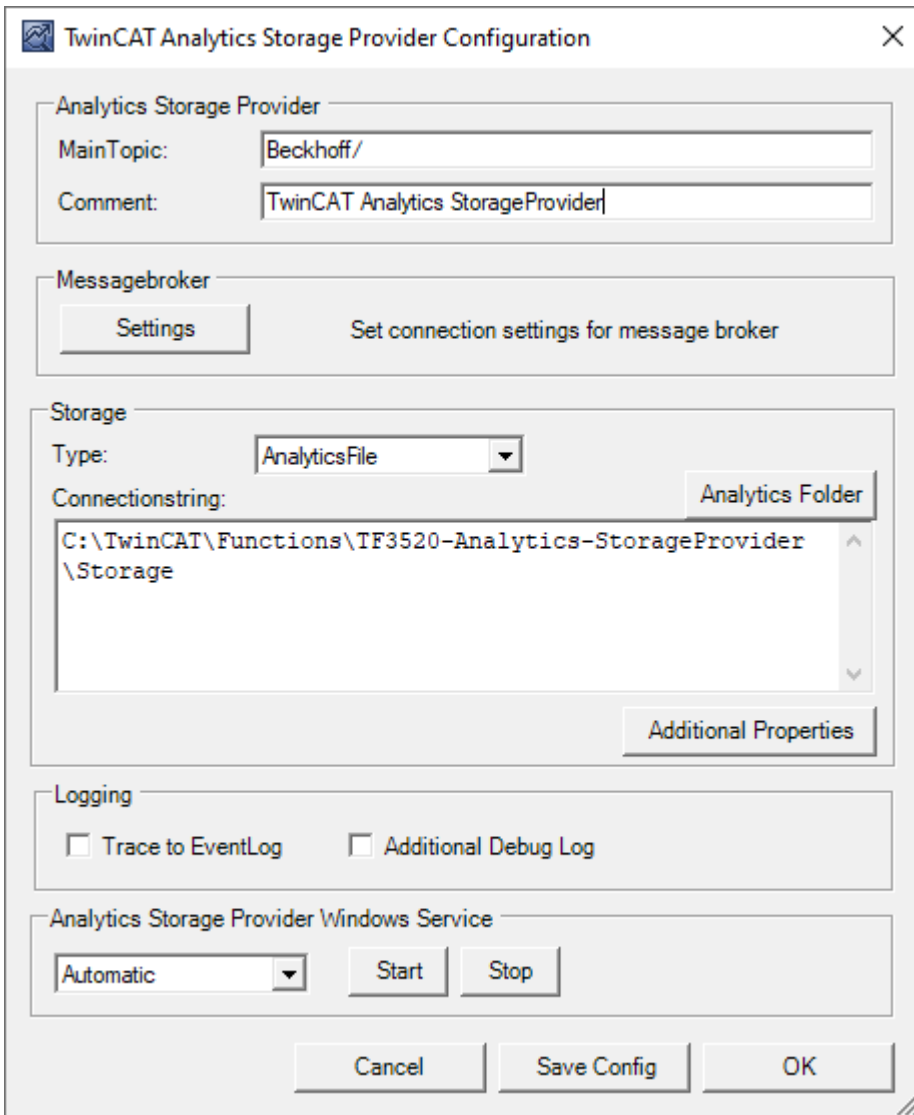
Currently, the Analytics workflow is fully mappable via MQTT. The engineering tools can also access the data of the machines via ADS and carry out analyzes.



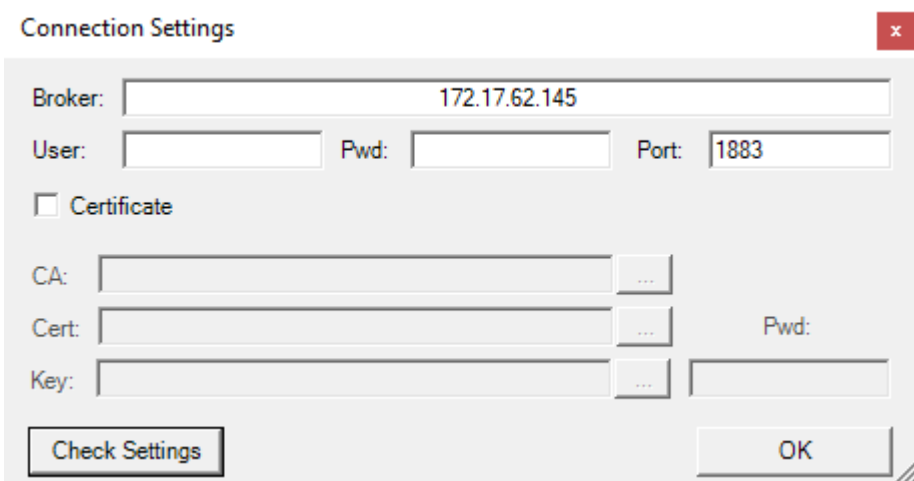
If you choose for the IoT communication protocol MQTT you have to setup a native MQTT Message Broker somewhere in the network (VM in a cloud system is also possible). This Message Broker provides a decoupling of the different applications in the Analytics Workflow.

### 4.3 Historicize data

After the TwinCAT Analytics Storage Provider has been installed, the service running in the background can be configured. For this purpose you can find the TcAnalyticsStorageProvider\_Config application in the C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\WinService folder.



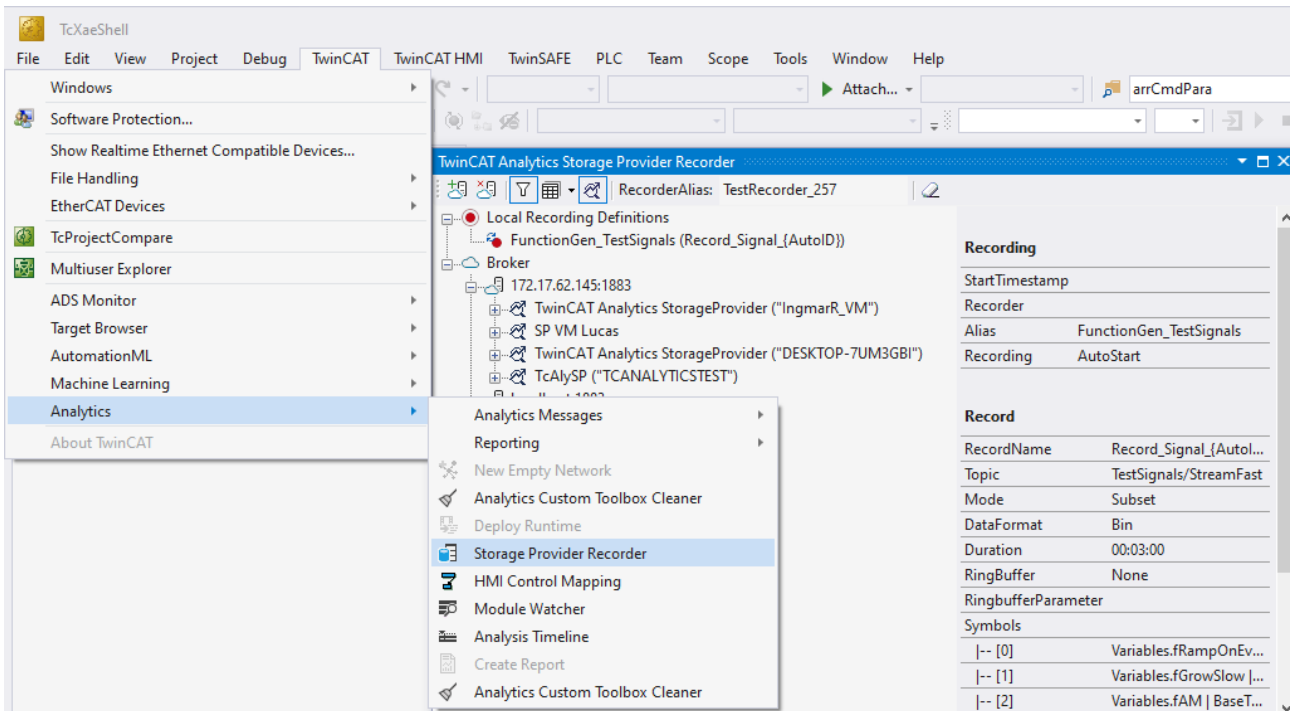
The main part of the topic can be defined in the configuration as well as the comment, which is used for identification if more than one Storage Provider is registered with the message broker.



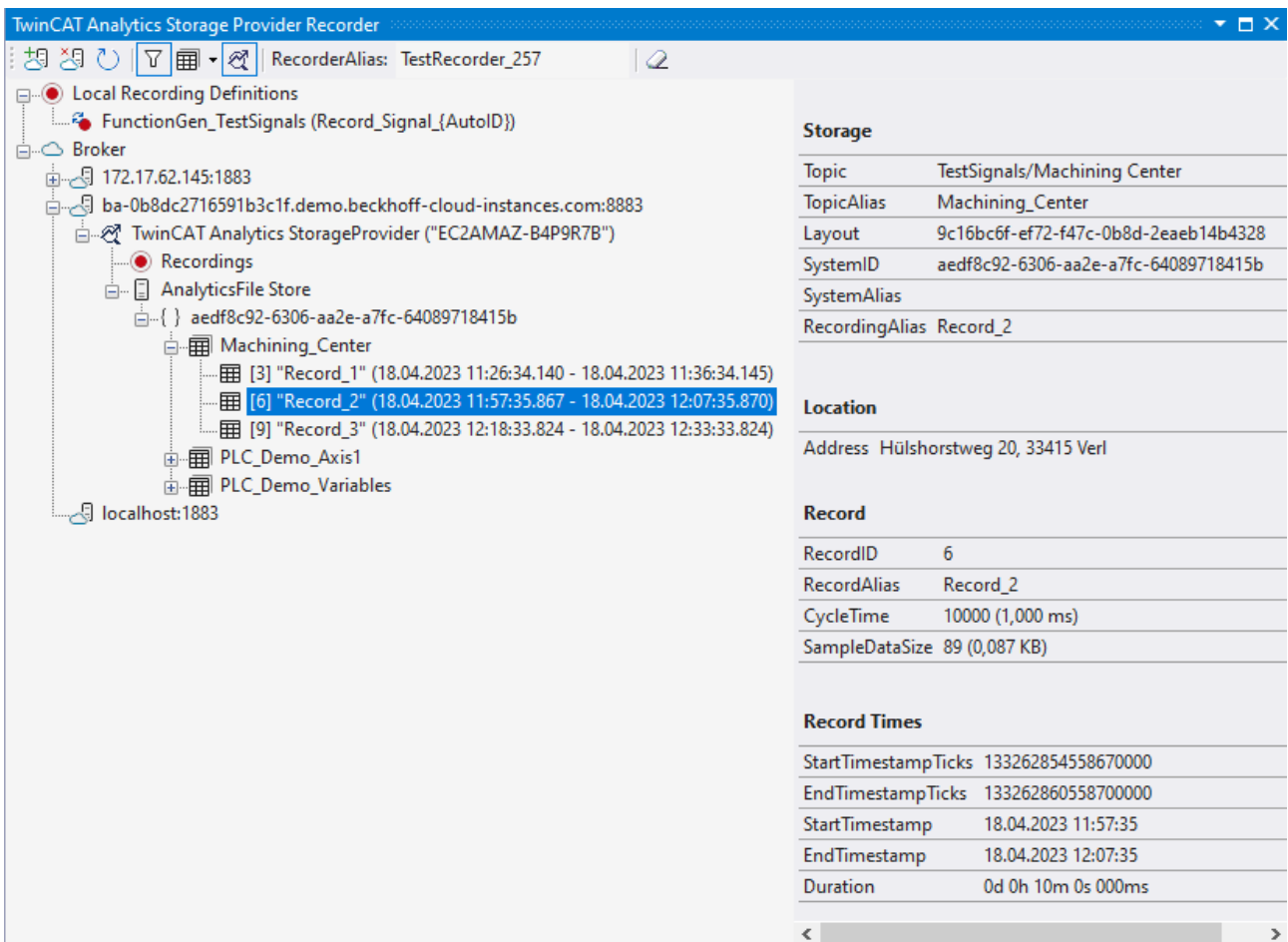
You can make the message broker settings and decide on a storage type:

- Analytics File (binary file)
- Microsoft SQL (binary)
- Microsoft Azure Blob (Azure Cloud required)

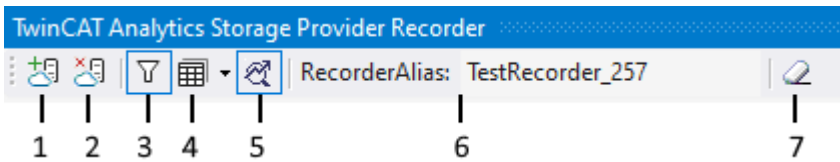
At last you can save the configuration and start the service. The next step is to configure the specific recording. For this you should select the **Storage Provider Recorder** in your development environment.



With the Storage Provider Recorder recording definitions can be created, started and managed. In addition, it is possible to manage the data memories of individual Analytics Storage Providers. All important properties of the found Analytics Storage Providers and historized data are clearly displayed.



**Toolbar**

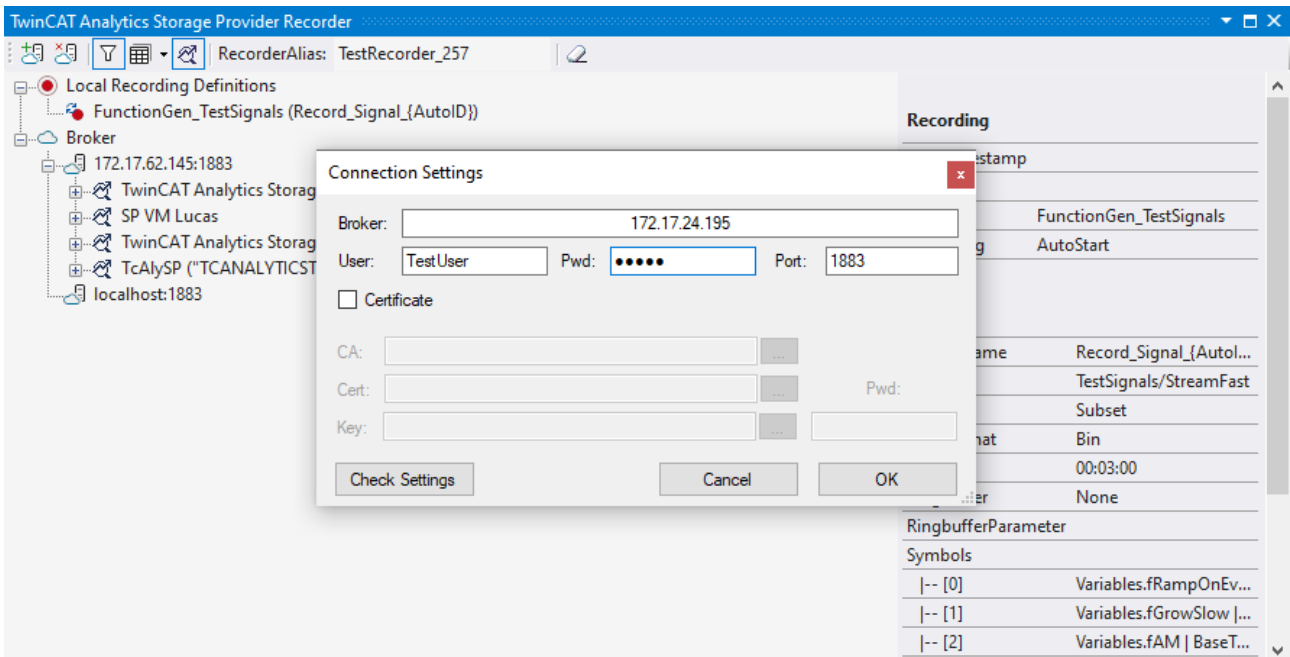


1	Add new broker
2	Remove selected broker
3	Display filters (All / Show my active recordings)
4	Select display type (Alias / Topics)
5	Show / Hide Offline Analytics Storage Provider
6	"RecorderAlias" - grouping name of recordings
7	Remove messages from error list

### Recorder window setup

First assign a "RecorderAlias". This helps to group the started recordings and to find its self started ones again. The filter can also be used to display recordings started by others.

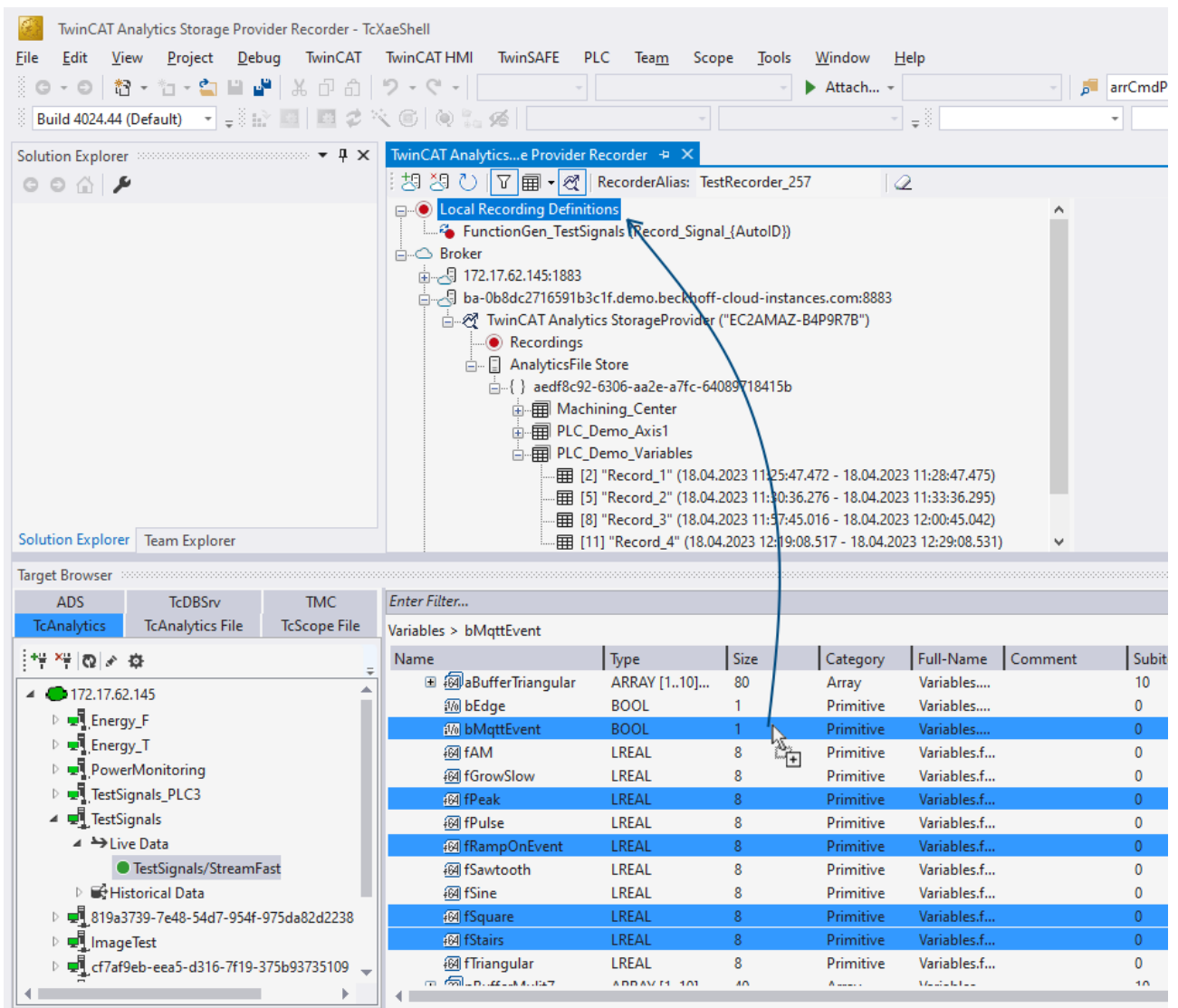
After that, one or more brokers can be set up. This is done via the already known input mask for MQTT connection properties.



Once a connection to the broker could be established, all Analytics Storage Providers connected to it will be listed.

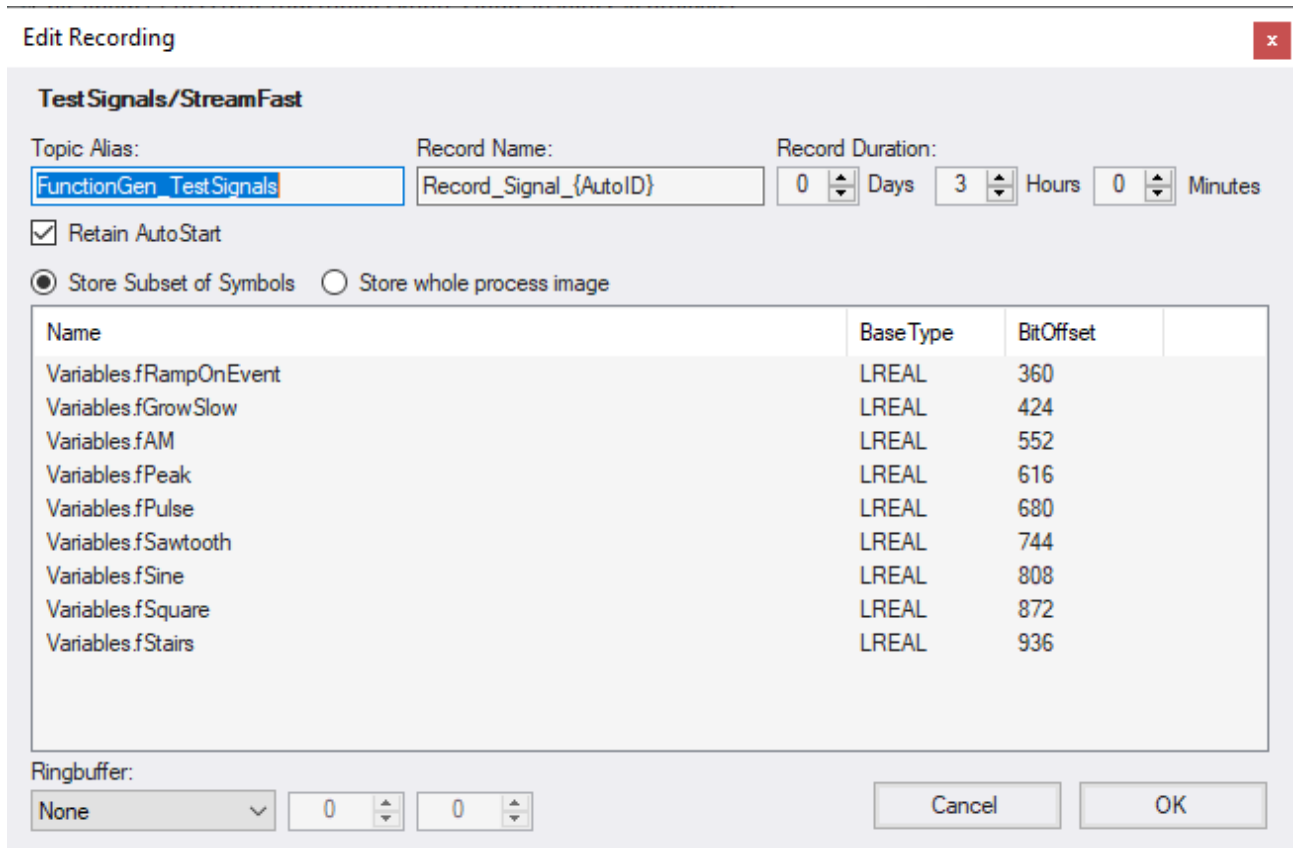
### Create recording definition

To configure the recording, select your target in the Target Browser. Click **Live Data** and select one or more variables by multiple selection and drag and drop them to the **Local Recording Definitions** node in the Recorder window.



In the recorder you can add the selected variables or the complete source process image of the variables.



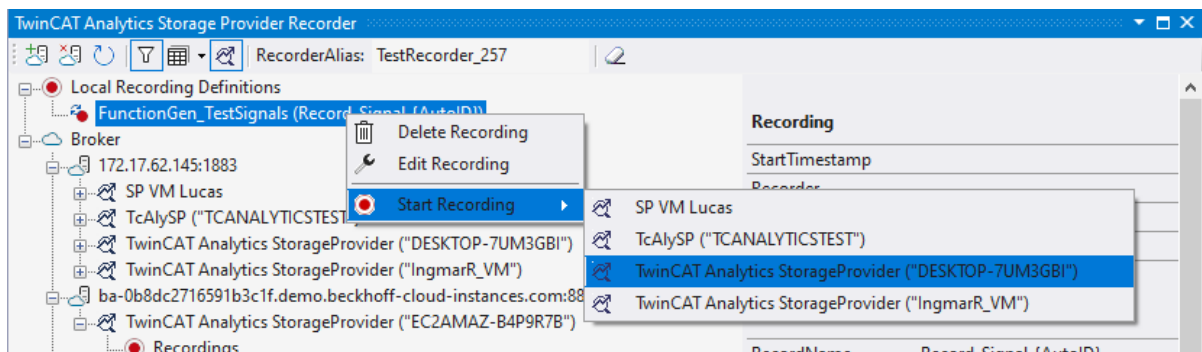


You can also configure recording names and a duration (otherwise the recording will run endlessly until it is stopped manually). A ring buffer can be set by storage or time.

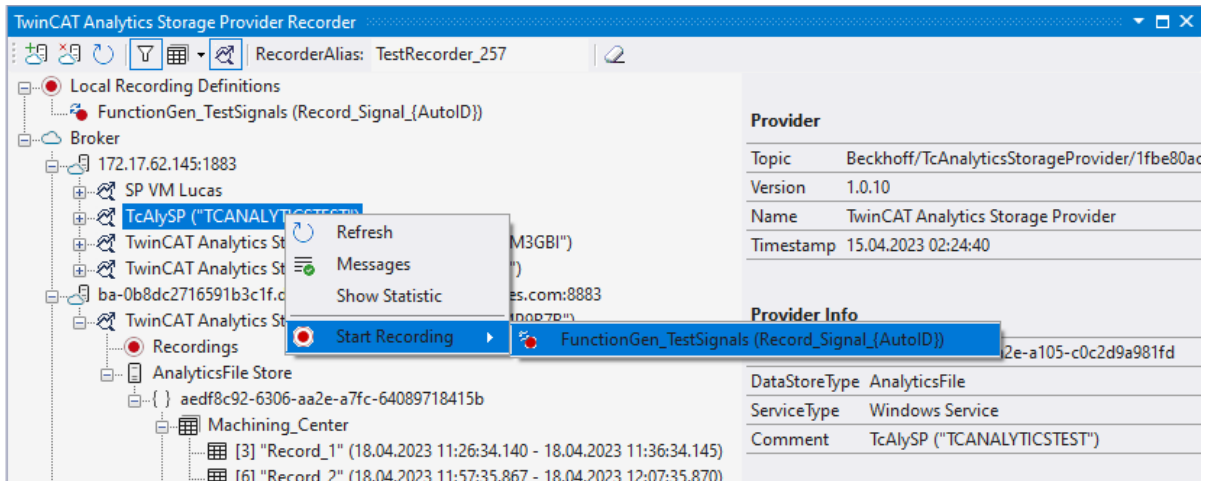
**Start recording**

There are three different ways to start a recording.

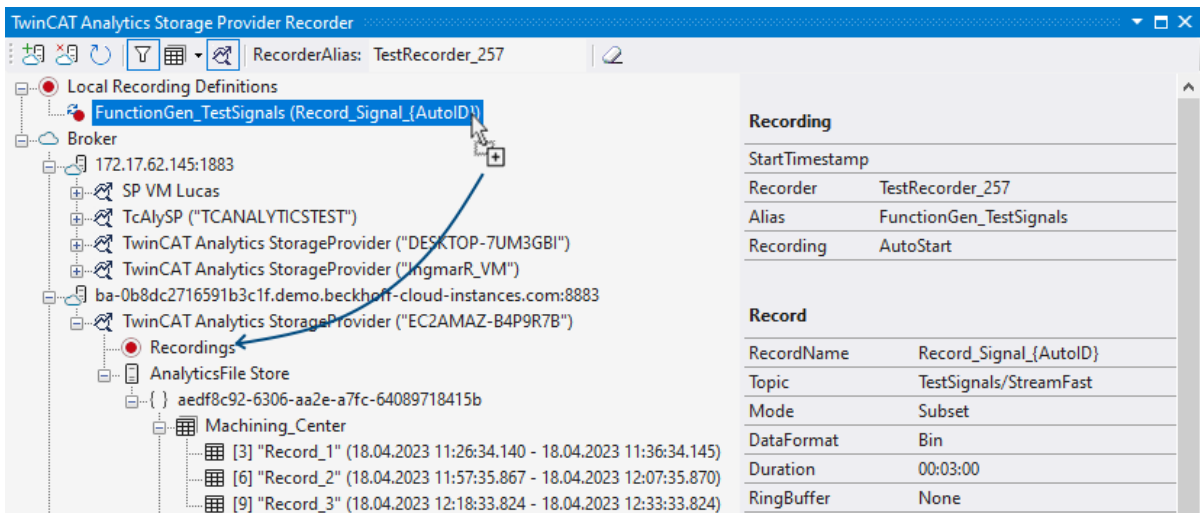
1. Via the context menu at a local recording definition node. Here you can select the desired Analytics Storage Provider, which should record the data. Only the storage providers that have access to the selected topic are displayed.



- From the context menu on the Analytics Storage Provider node. The desired recording definition can be selected here. Only the definitions that can also be processed by the Analytics Storage Provider are displayed.

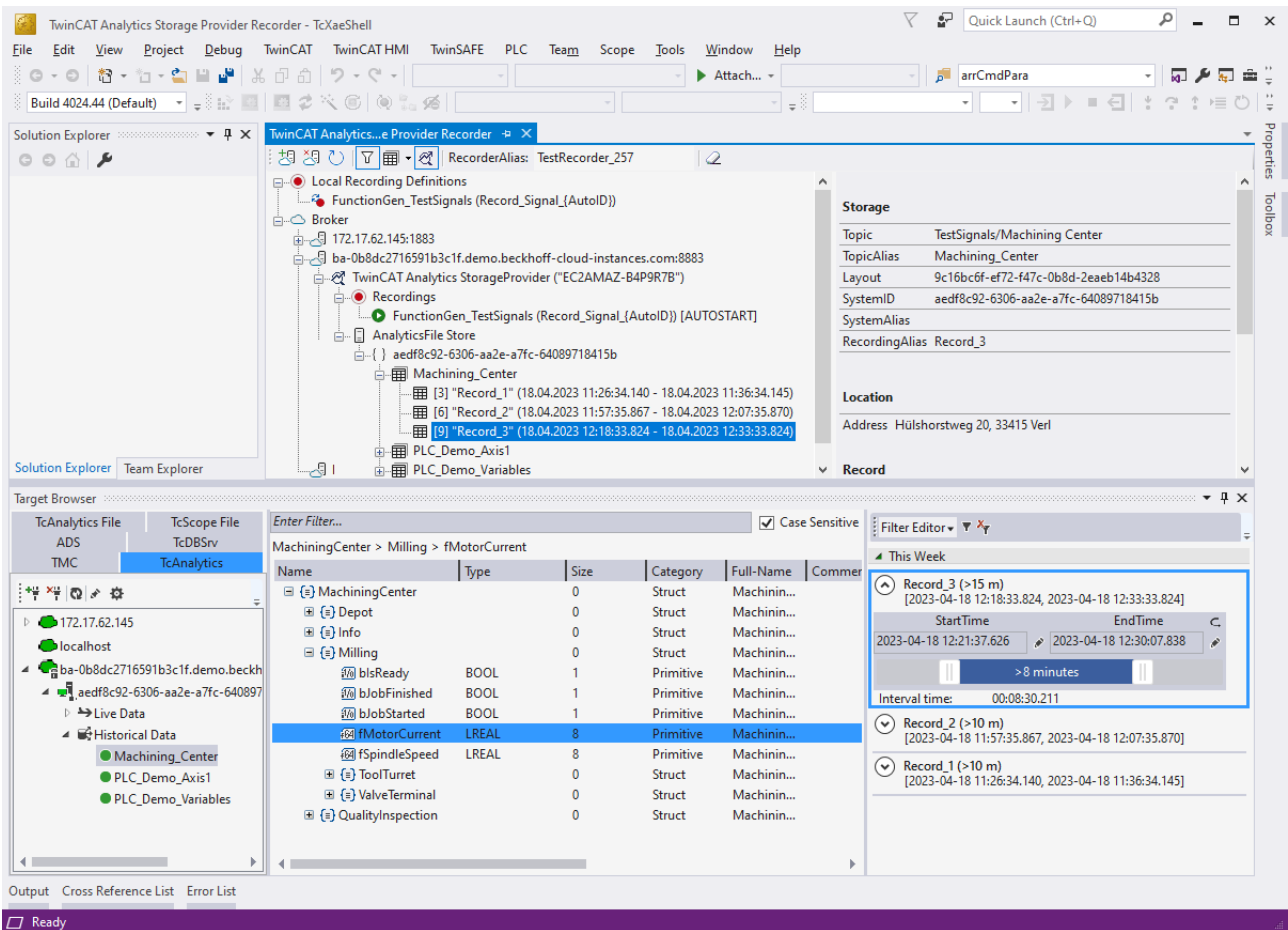


- Drag and drop the local recording definition node. The desired recording definition can be started on the desired Analytics Storage Provider via drag-and-drop. If the Storage Provider is unable to execute the recording definition, an error message is displayed in the error list.



**Use historized data**

After and also during recording, you can select the historical data as input for your analysis in Target Browser. In the Target Browser, you will find a new control on the right side for the historical data. There you can select the timespan for your data.



## 4.4 Import Analytics Files

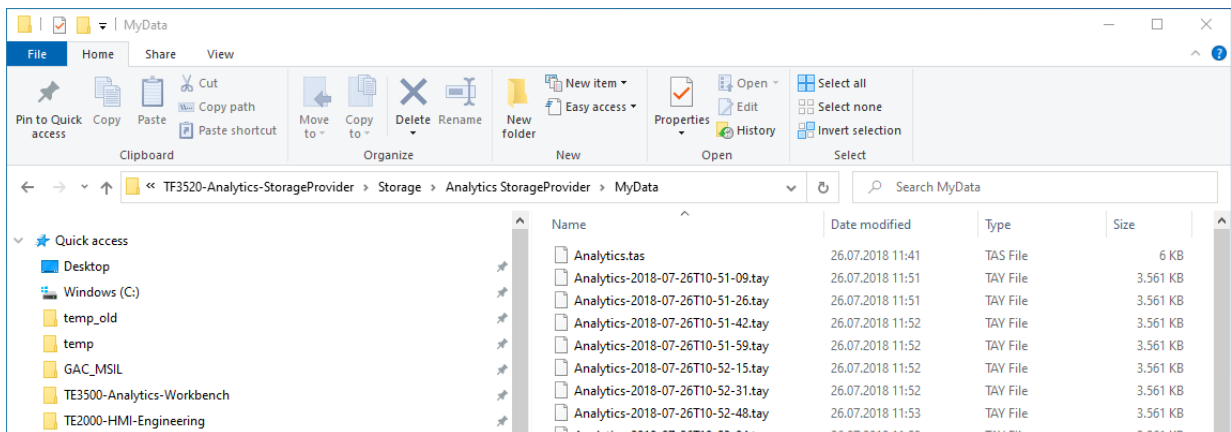


In the following it is assumed that you have installed TwinCAT under "C:/TwinCAT". Otherwise, you must adjust the specified paths accordingly.

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

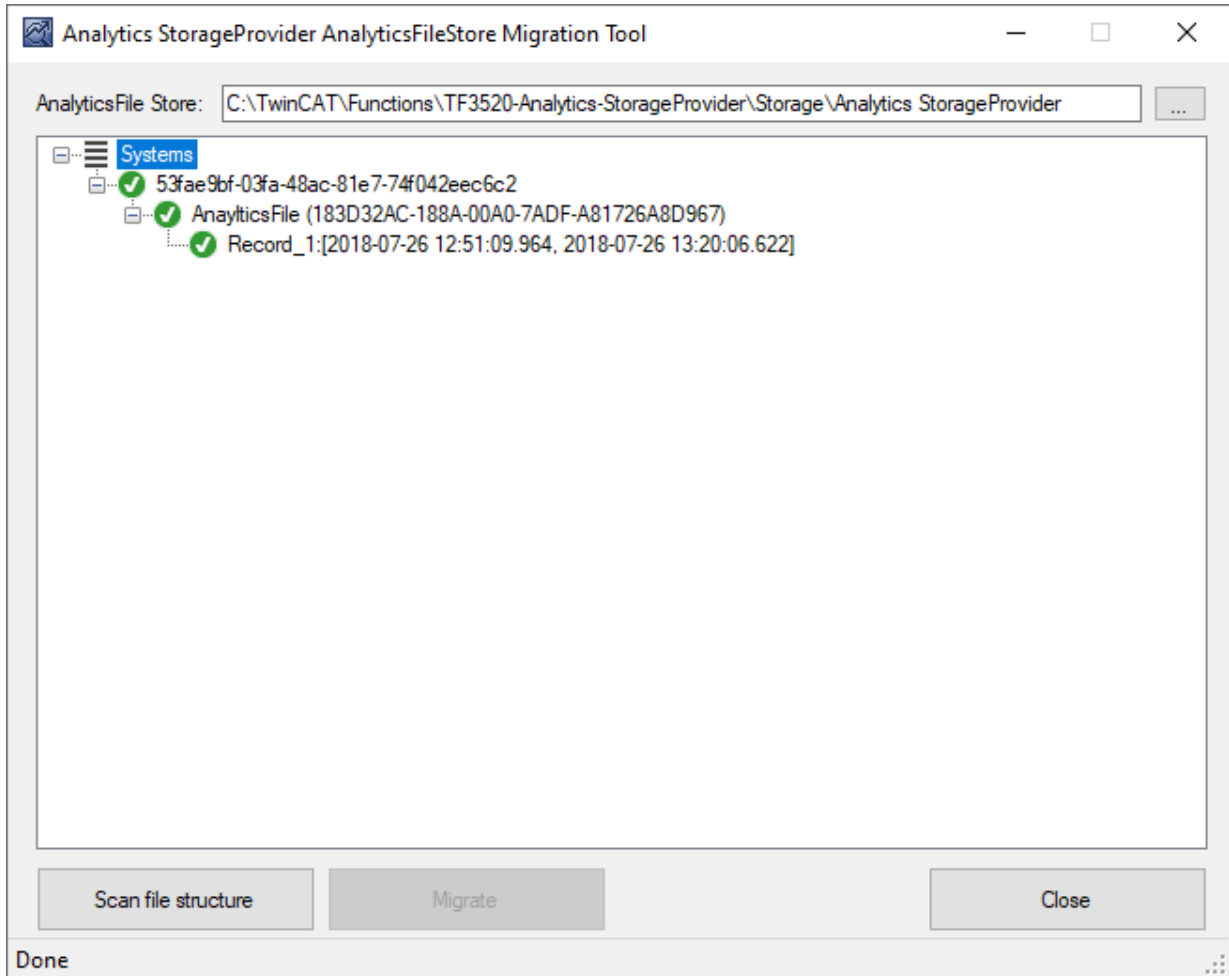
To do this, perform the following steps:

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 *TcAnalyticsSPAllyFileStoreMigration.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**.



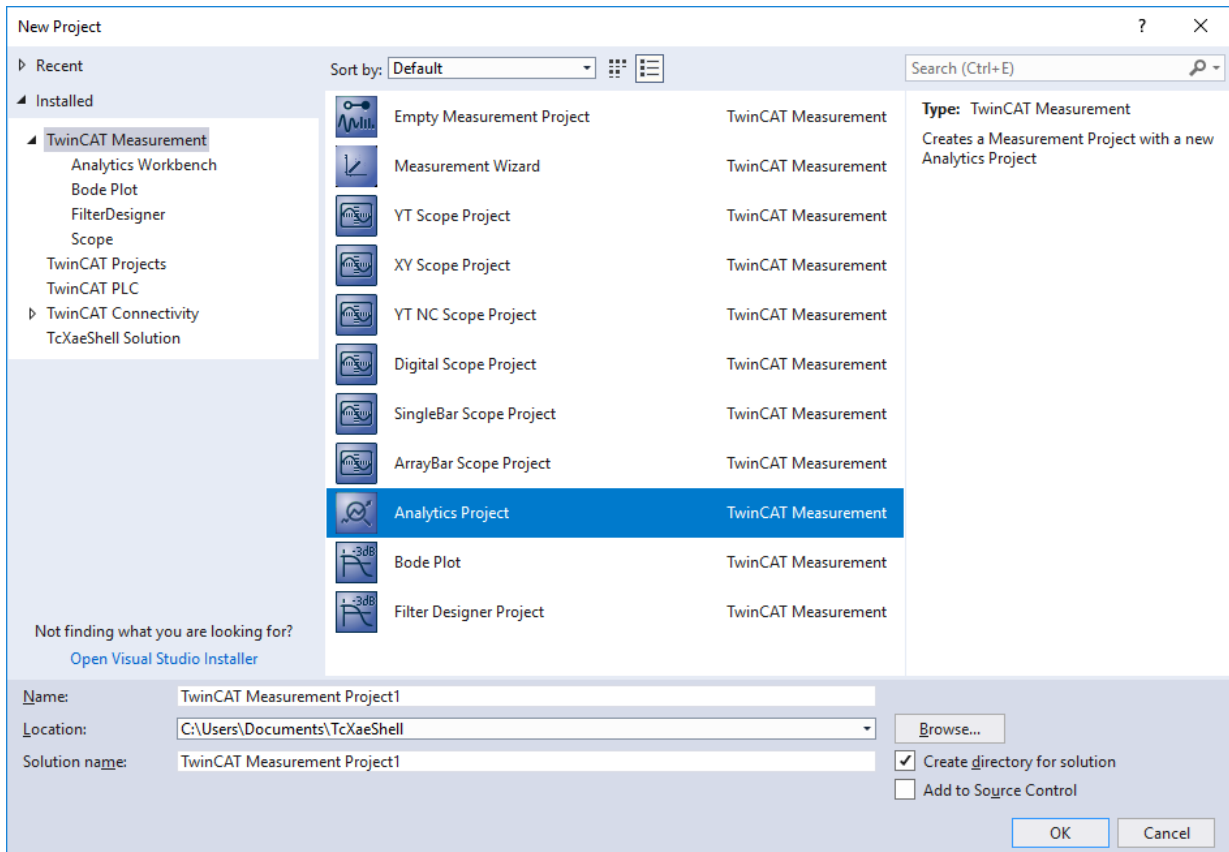
6. Now you can see your imported data in the [TwinCAT Target Browser \[► 55\]](#).

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

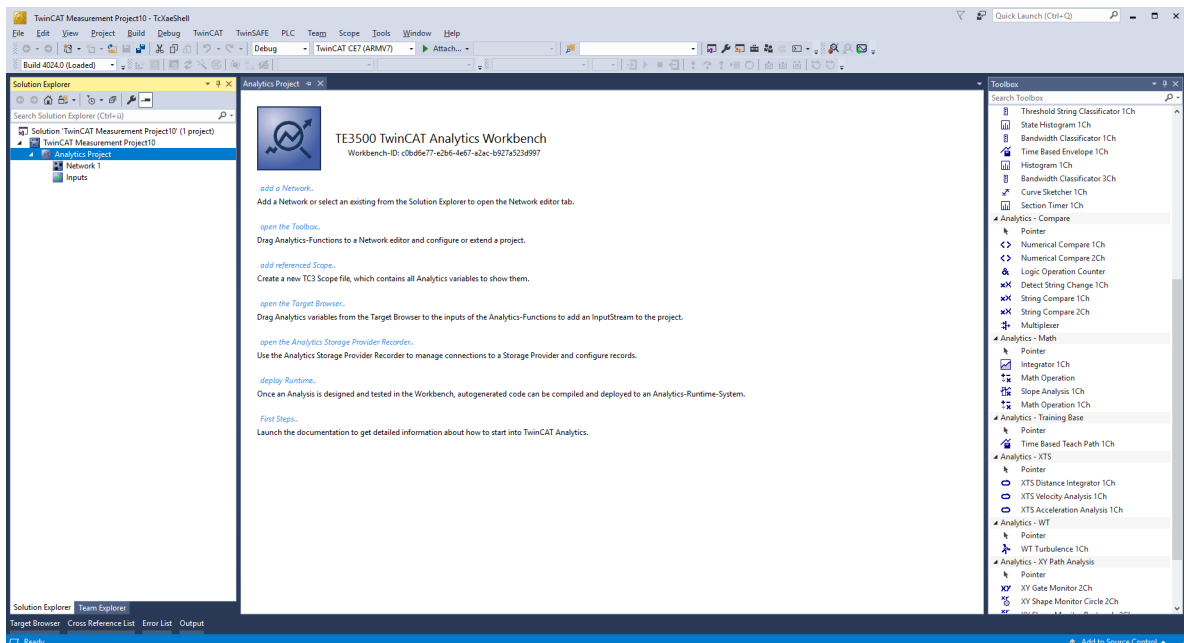
## 4.5 Analyse data

- ✓ Open your TwinCAT Engineering environment to start the data analysis.
1. Open **Visual Studio® > File > New > Project...**

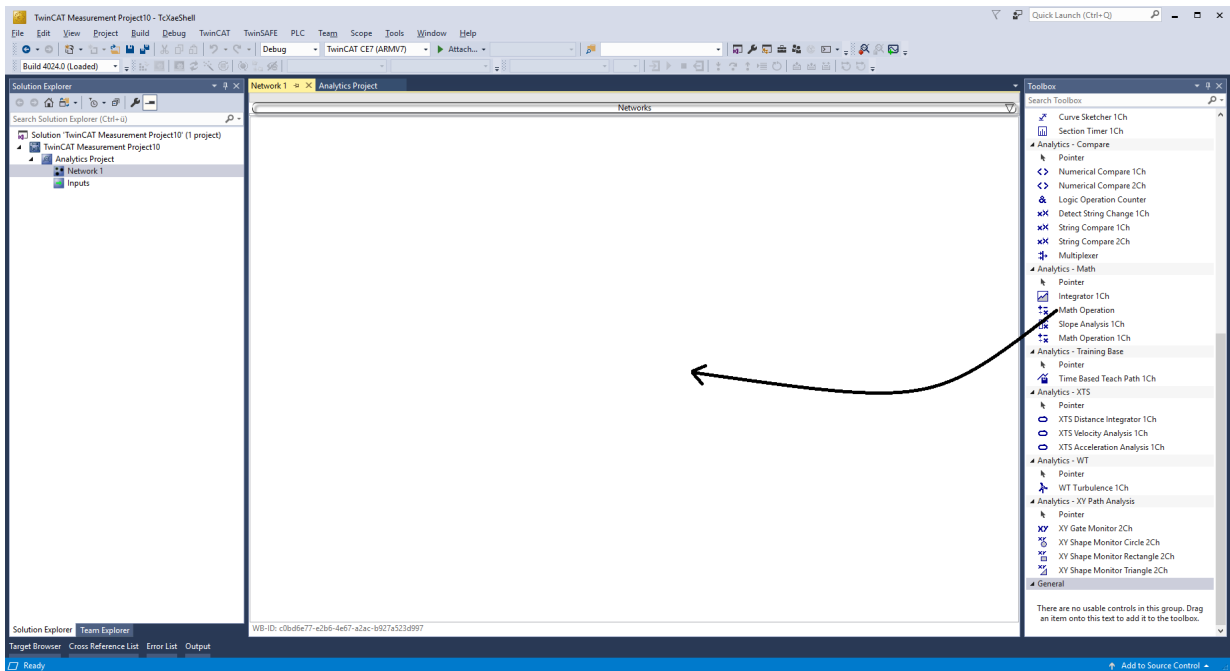
2. Select the **Analytics project template** from **TwinCAT Measurement**.



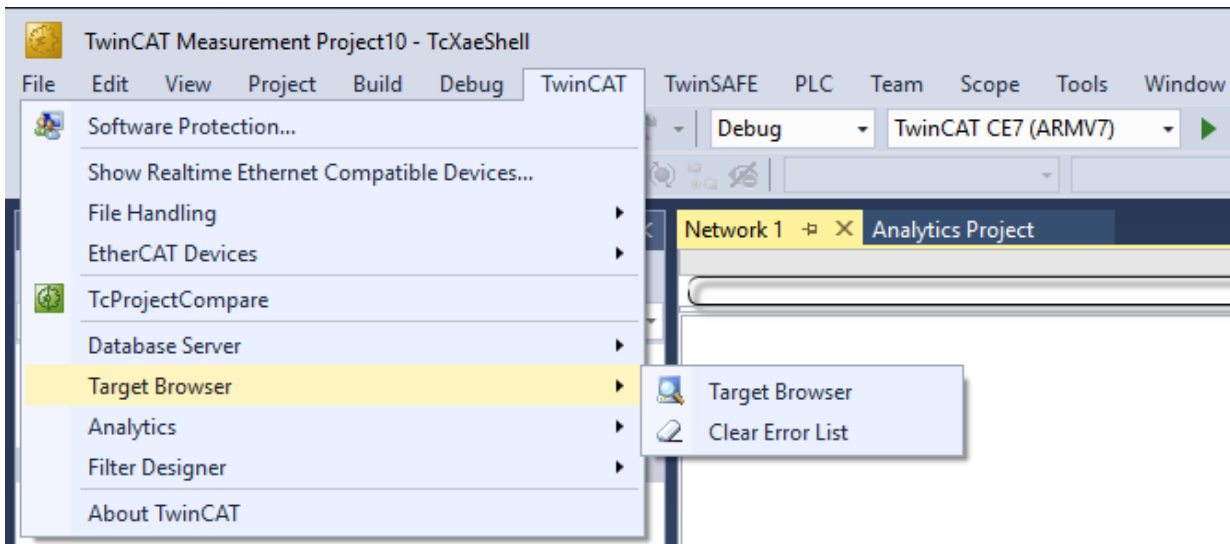
⇒ The new project is displayed in the Solution Explorer. After clicking the **Analytics Project** tree node element a start window opens where you can select your first action. From here you can add a network, open the **Toolbox**, open the **Target Browser** or open the **Analytics Storage Provider 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.

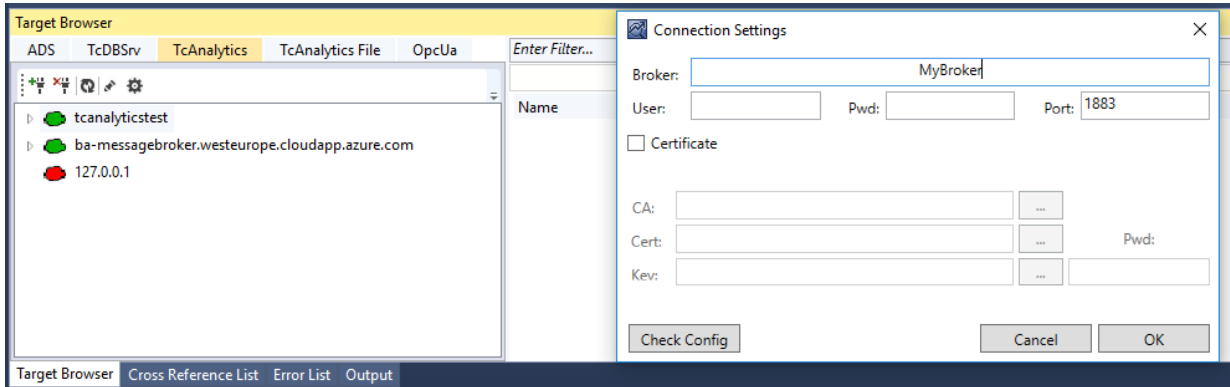


- When you click on the network, an editor opens. Now you can drag and drop the desired algorithm into the editor interface.
- After selecting the algorithm, you need to connect input variables to the modules (algorithm). To do this, open the **Target Browser**.  
**TwinCAT > Target Browser > Target Browser**

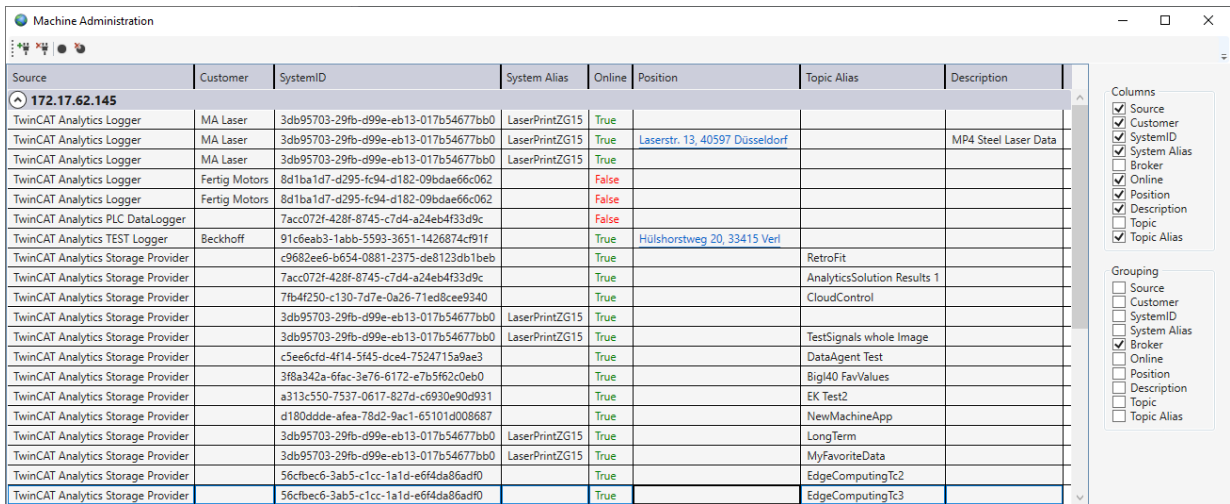


- Now select the **TcAnalytics** or **TcAnalyticsFile** tab in the Target Browser. Continue with the tab **TcAnalytics** (MQTT).

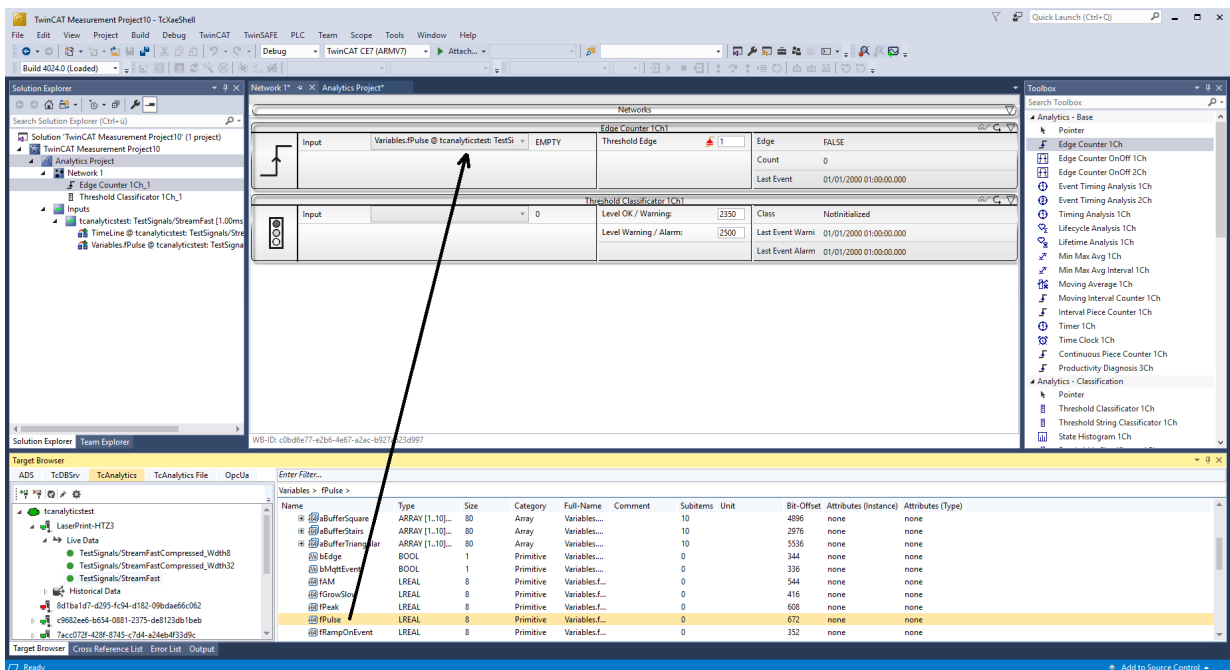
- Click the icon highlighted in green in the toolbar of this Analytics extension. A window opens in which you can specify the connectivity data of your message broker.



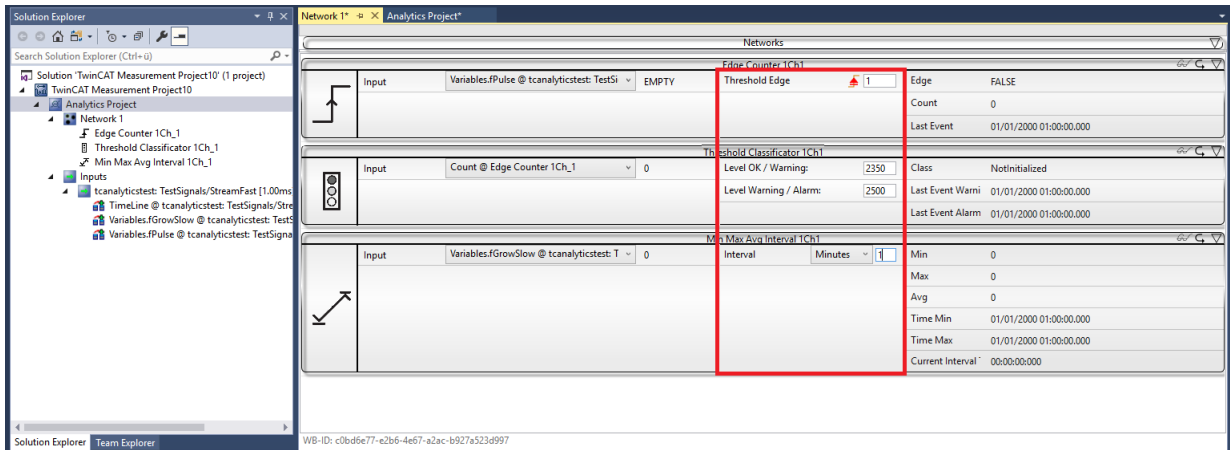
- Select your MQTT Analytics client (TwinCAT Analytics Logger, TwinCAT IoT Data Agent or Beckhoff EK9160). There is a unique ID for each control. This ID is displayed in the Target Browser.
- Clicking on the **gear icon**, you will get to the Machine Administration page. Here you can assign a system alias name that will be displayed in the Target Browser instead of the ID.



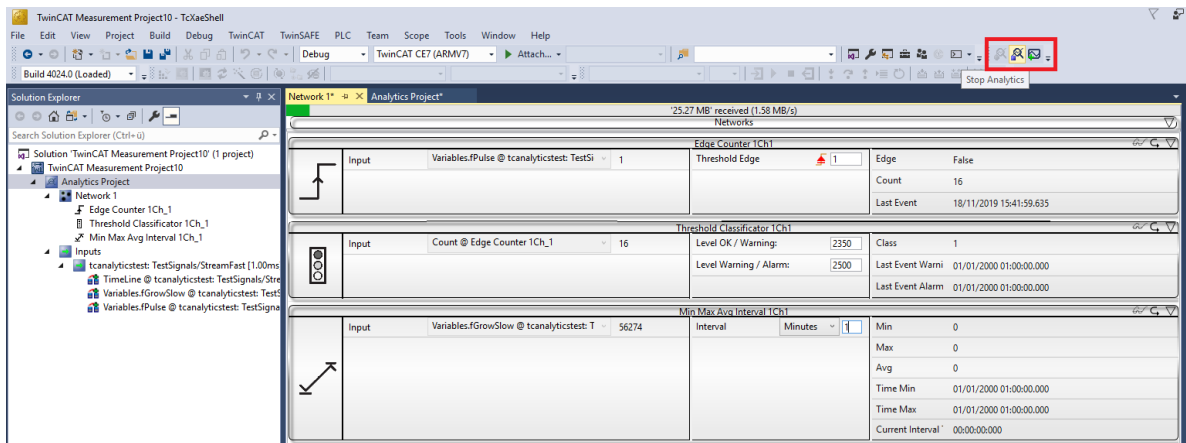
- In the next step, you can choose between live data and historical data for each MQTT Analytics client. In this case, the historical data is provided by the TwinCAT Analytics Storage Provider.



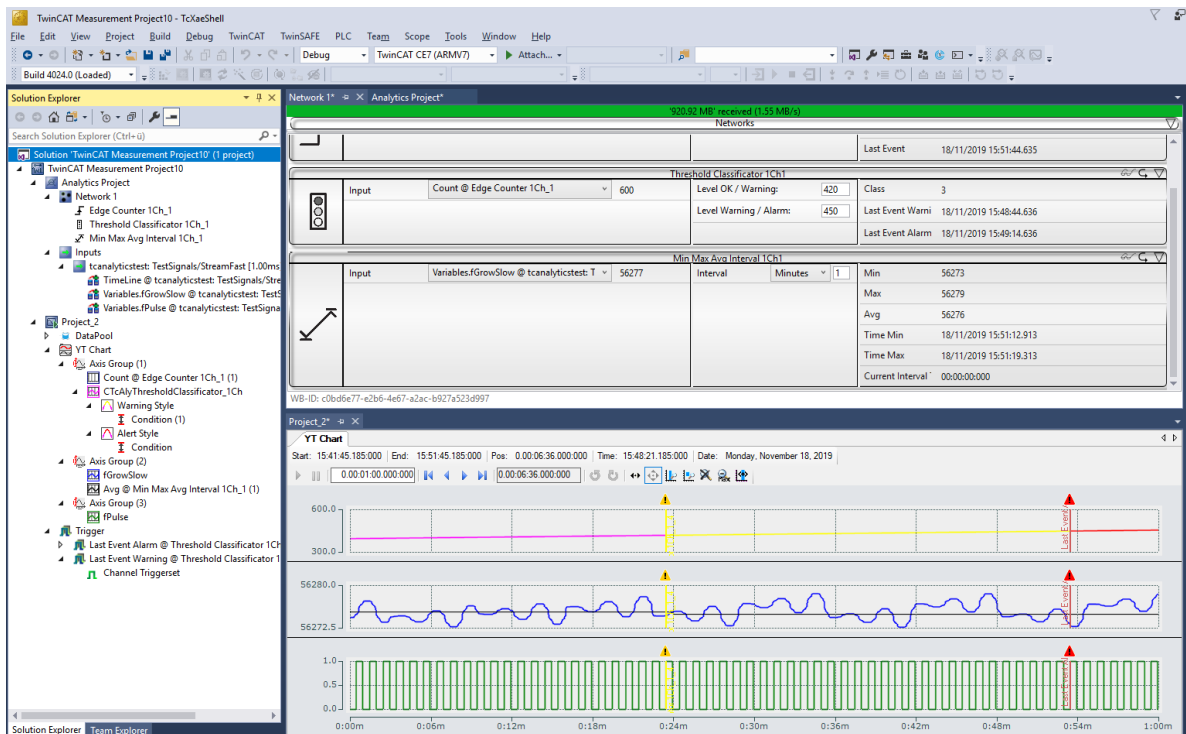
11. You can drag and drop the variables into the inputs of the specific algorithm. In most algorithms, conditions such as thresholds, time intervals, logical operators etc. can be specified. These settings are made in the middle of each module.



⇒ 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.



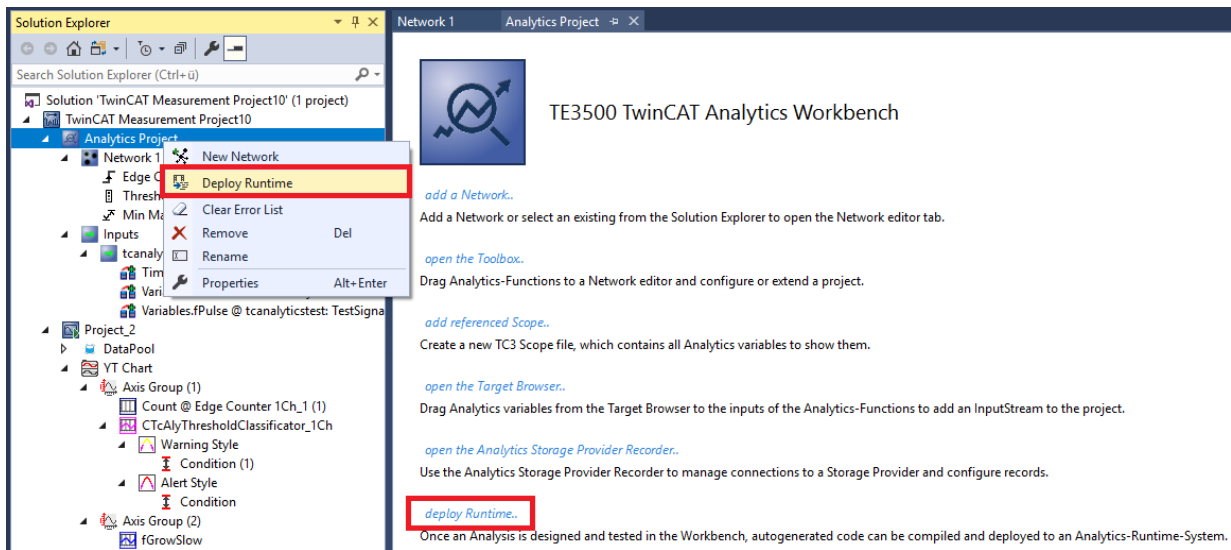


- ⇒ The analysis results can be displayed in the Scope View graphs using drag-and-drop. For example, a mean value can be displayed as a new channel in the view. Timestamps as markers on the X-axes show significant values.

## 4.6 24h Analytics application

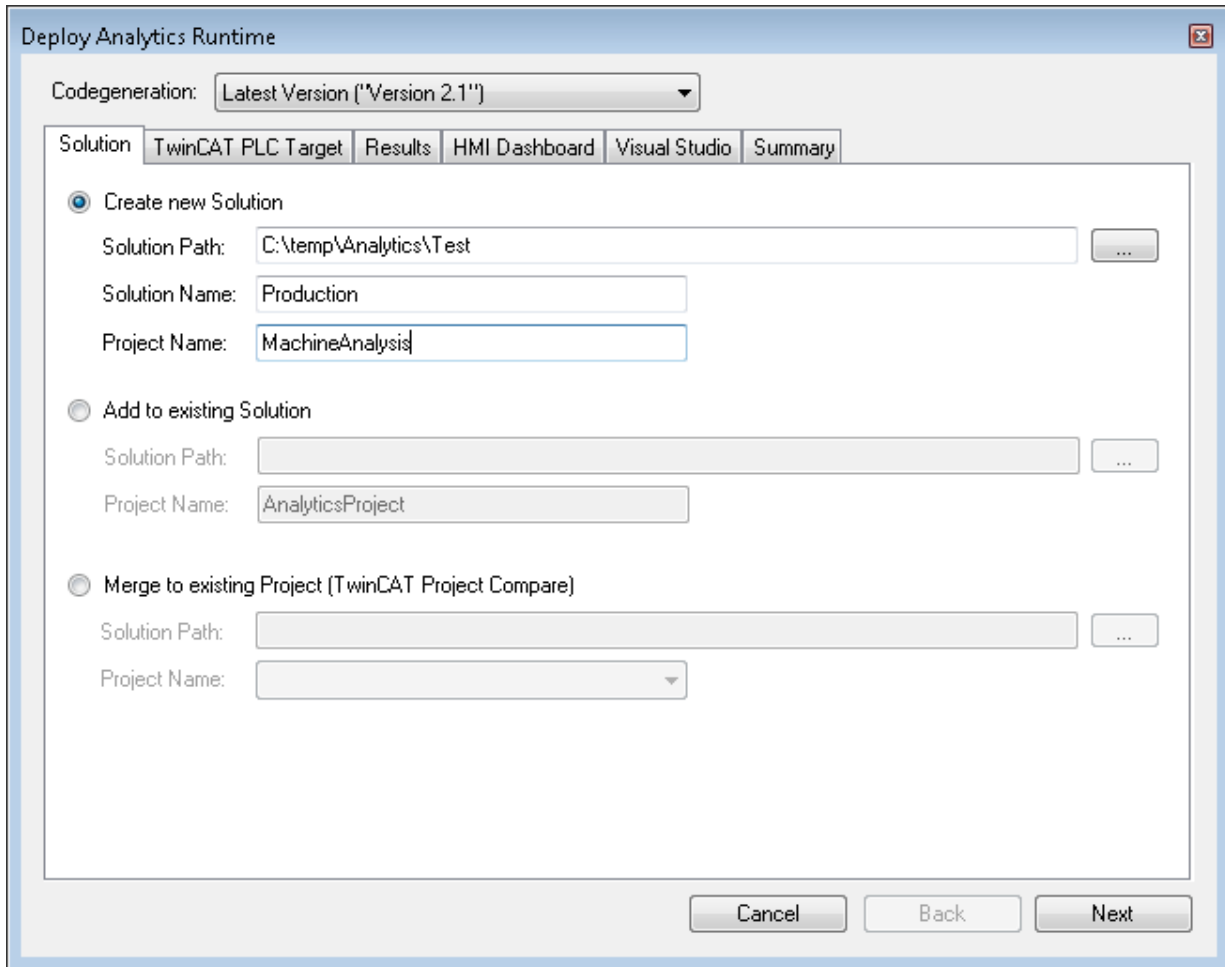
The last major step in the TwinCAT Analytics workflow is the continuous 24-hour machine analysis. It runs in parallel with the machine applications in the field. To make this very easy, the TwinCAT Analytics Workbench can automatically generate PLC code and an HTML5-based dashboard of your Analytics configuration. Both can be downloaded into a TwinCAT Analytics Runtime (TC3 PLC and HMI Server) and provide the same analysis results as the configurator tool in the engineering environment.

- ✓ First, save your configuration and open the Analytics Deploy Runtime Wizard. This can be done from the context menu in the Analytics Project tree item or from the start page.

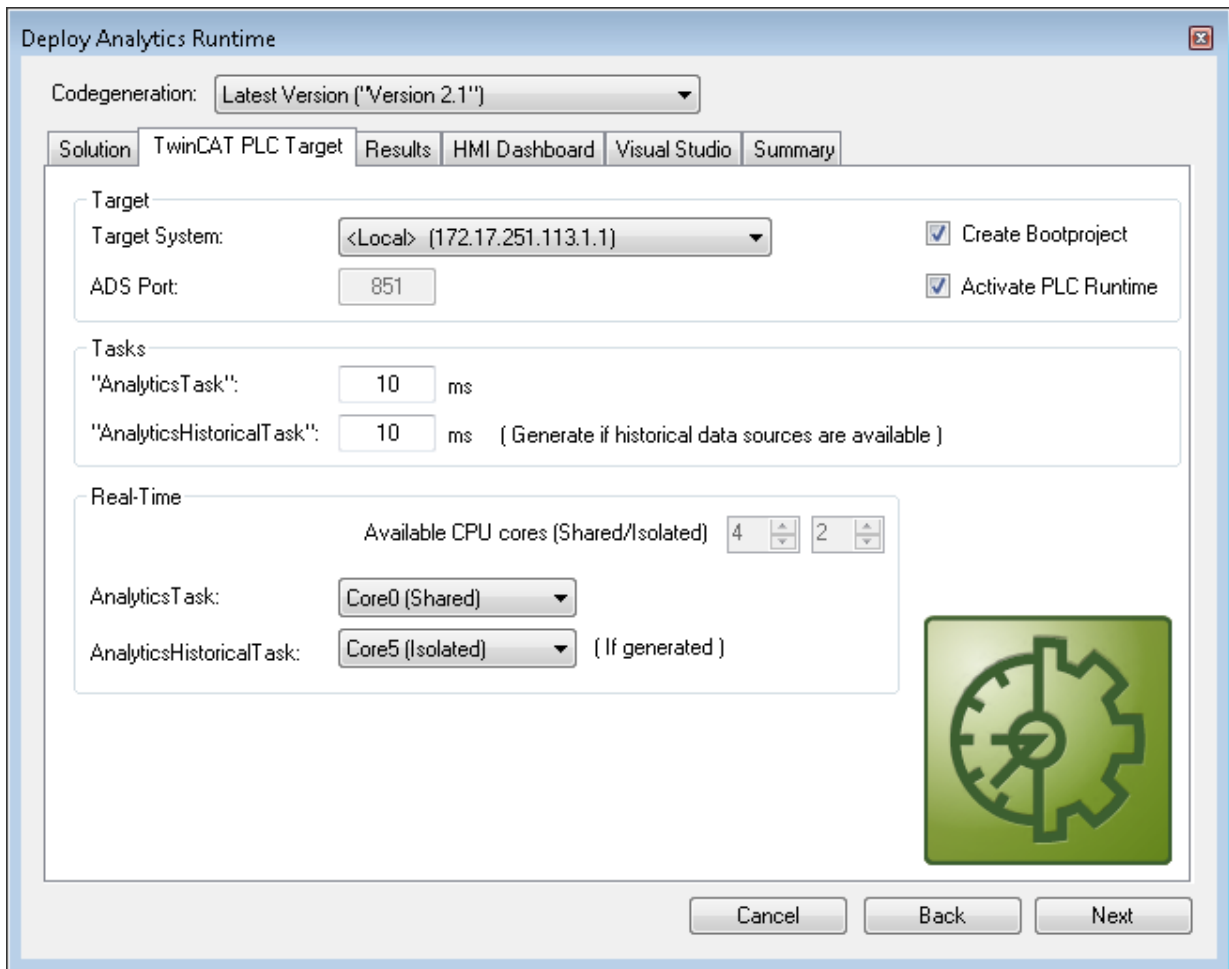


1. When the wizard is open, you can click through some tabs. The first one is called Solution. Here you can decide how your Analytics project should be used in the PLC code: As... completely new solution.

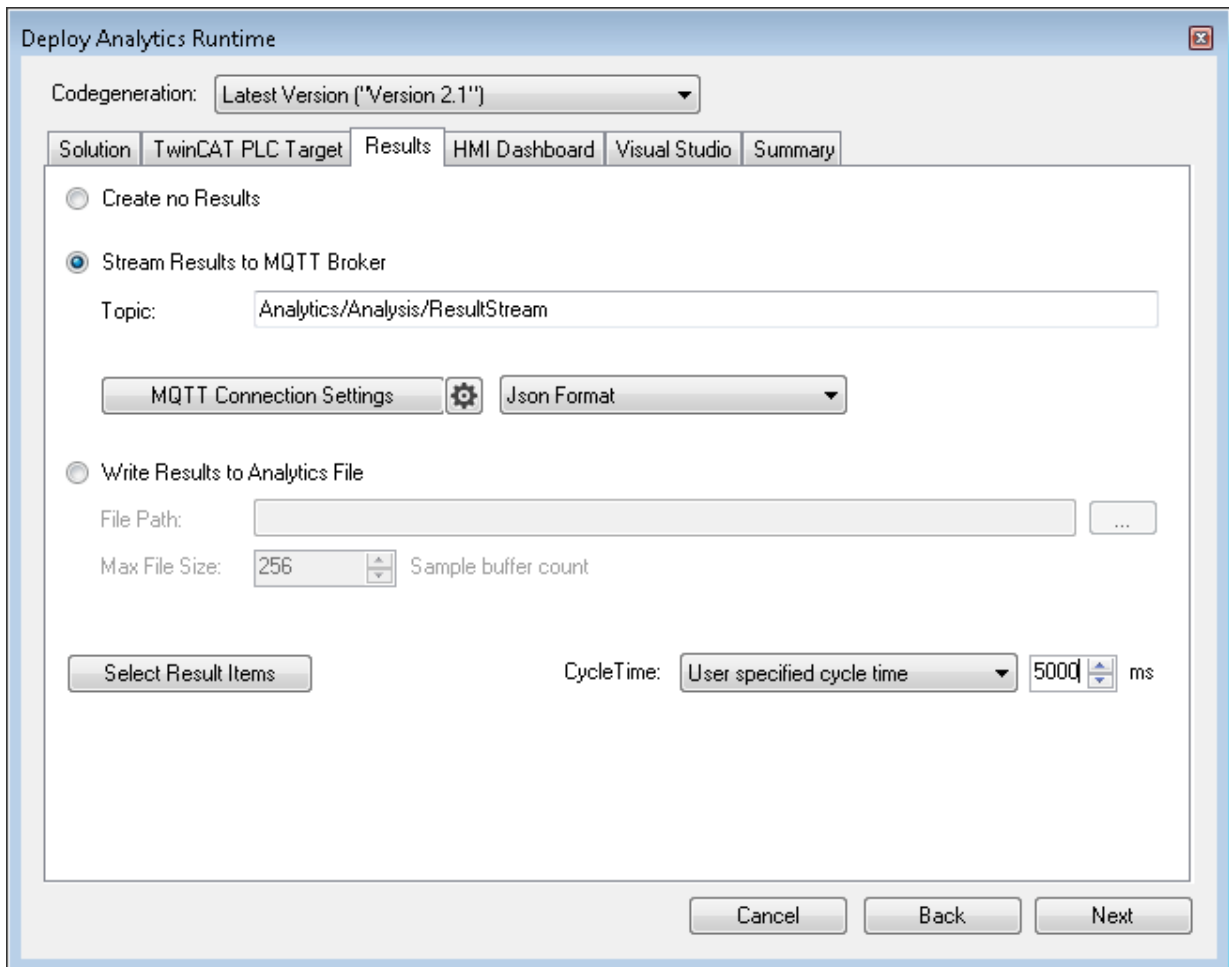
part of an existing solution.  
 update of an existing Analytics solution.



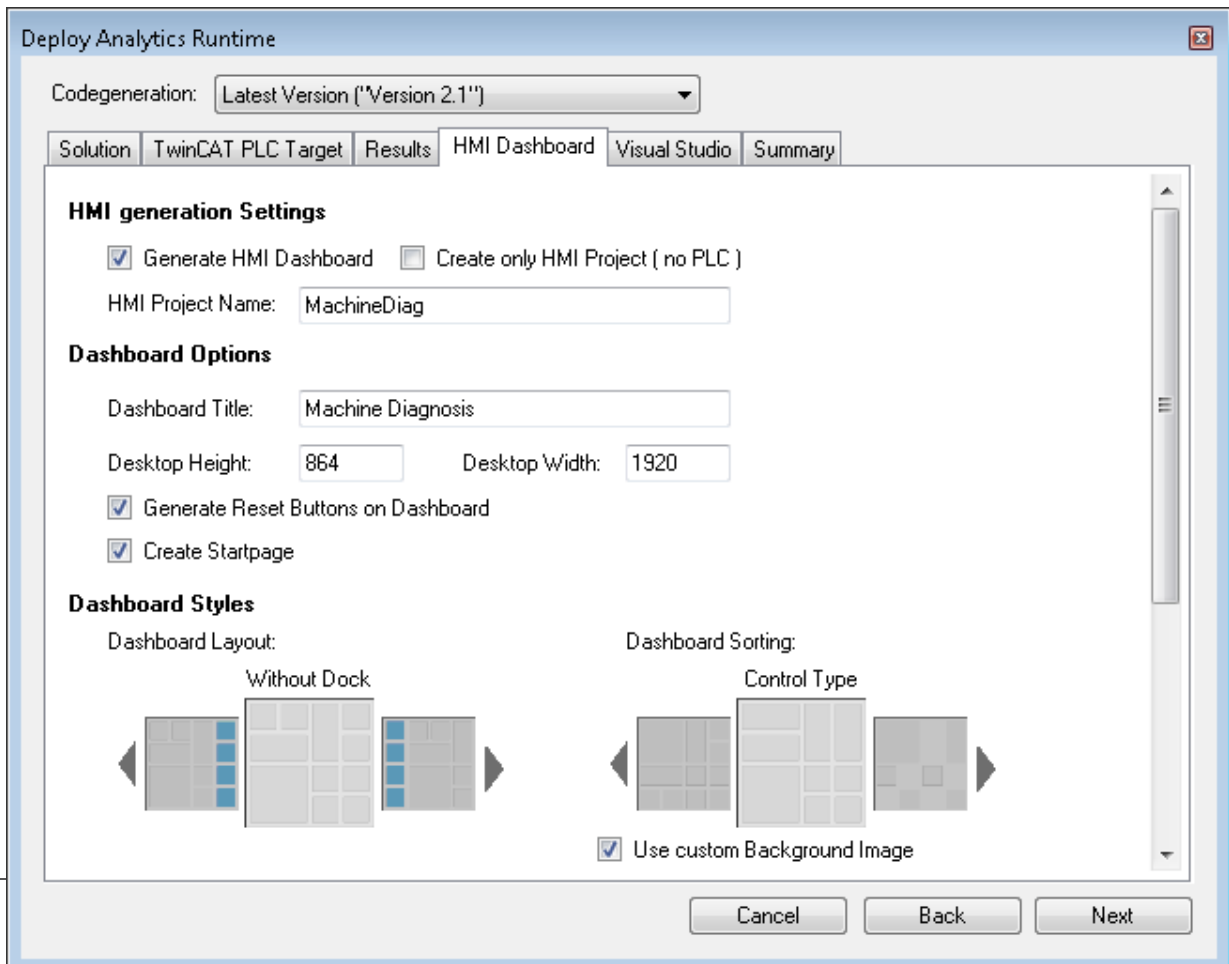
- In the **TwinCAT PLC Target** tab you can select the ADS target system that runs the TwinCAT Analytics Runtime (TF3550). The created project is immediately executable. For this purpose you can set the Activate PLC Runtime option. In addition, it can be selected that a boot project is created directly.



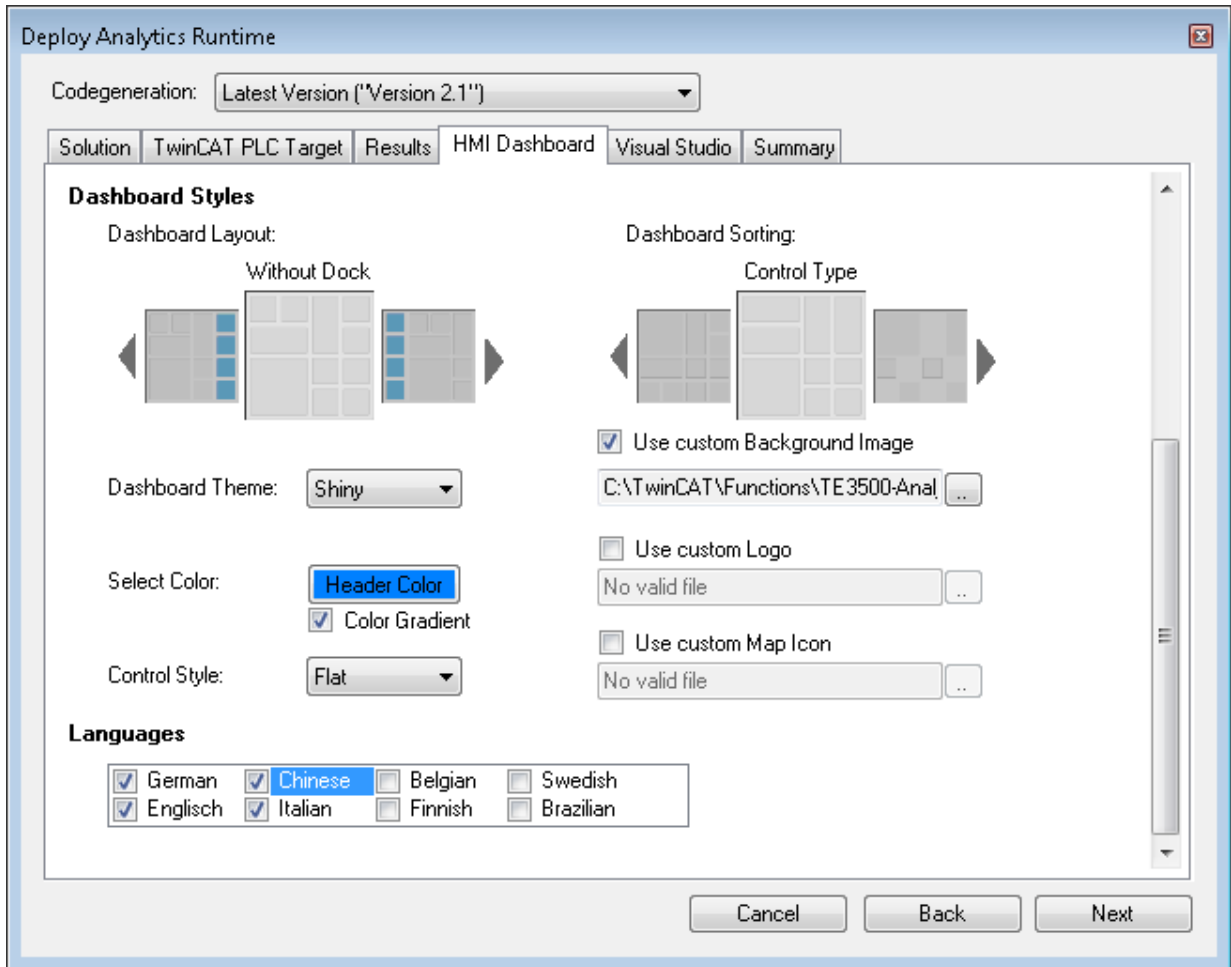
- Especially for virtual machines, it is important to run the project on isolated cores, which is also an option in this tab. The next tab **Results** is needed only if you have selected the **Stream Results** option in the algorithm properties. If you want to send results, you can decide here in which way (locally in a file/ through MQTT) and which format (binary/JSON) this should be done. This is also generated automatically and executed immediately after activation.



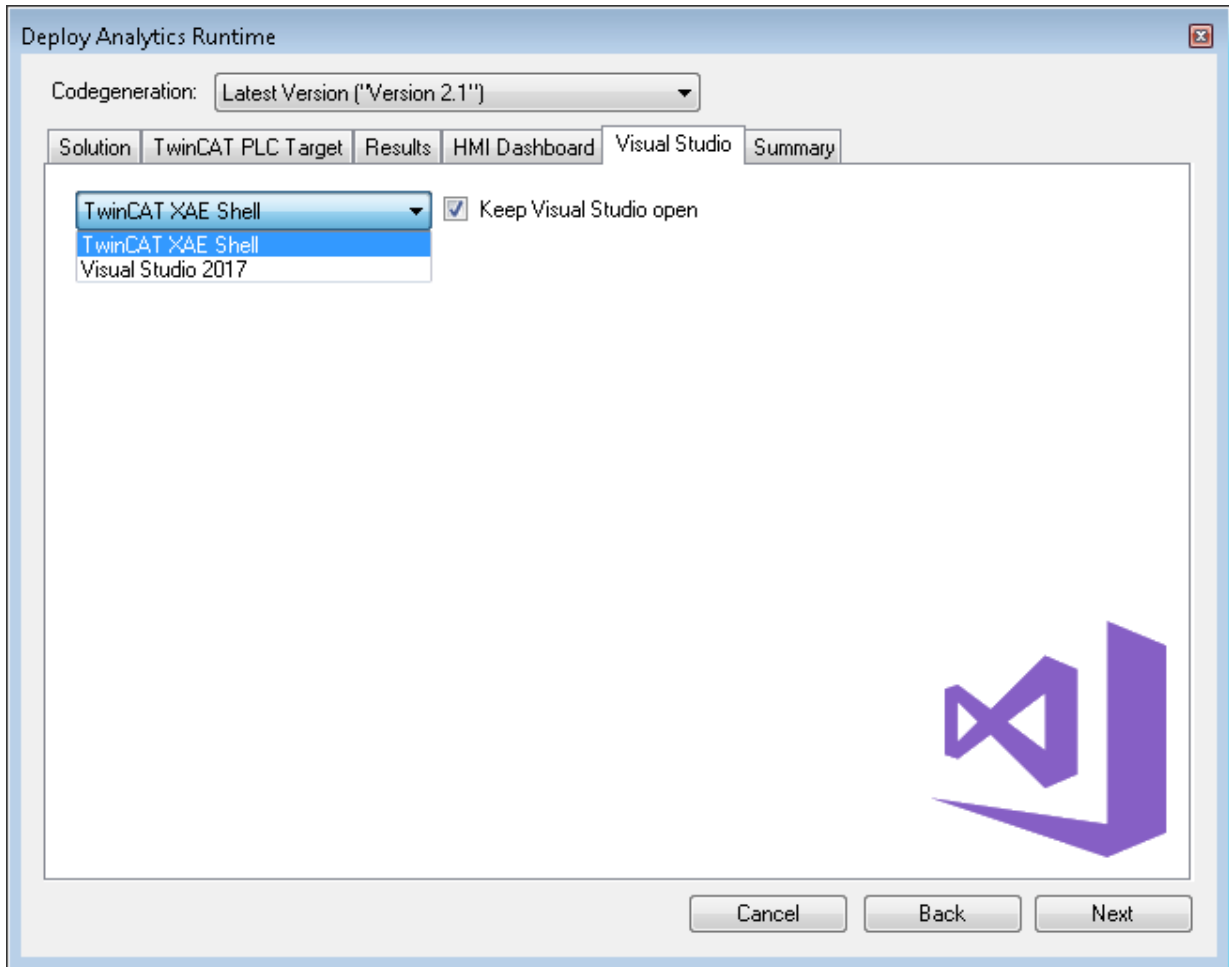
Downsampling of the results is possible by specifying a cycle time. The next tab is for the **HMI Dashboard**. A prerequisite for the automatic generation of the dashboard is the selection of HMI Controls for the corresponding algorithms whose results are to be displayed in the dashboard.



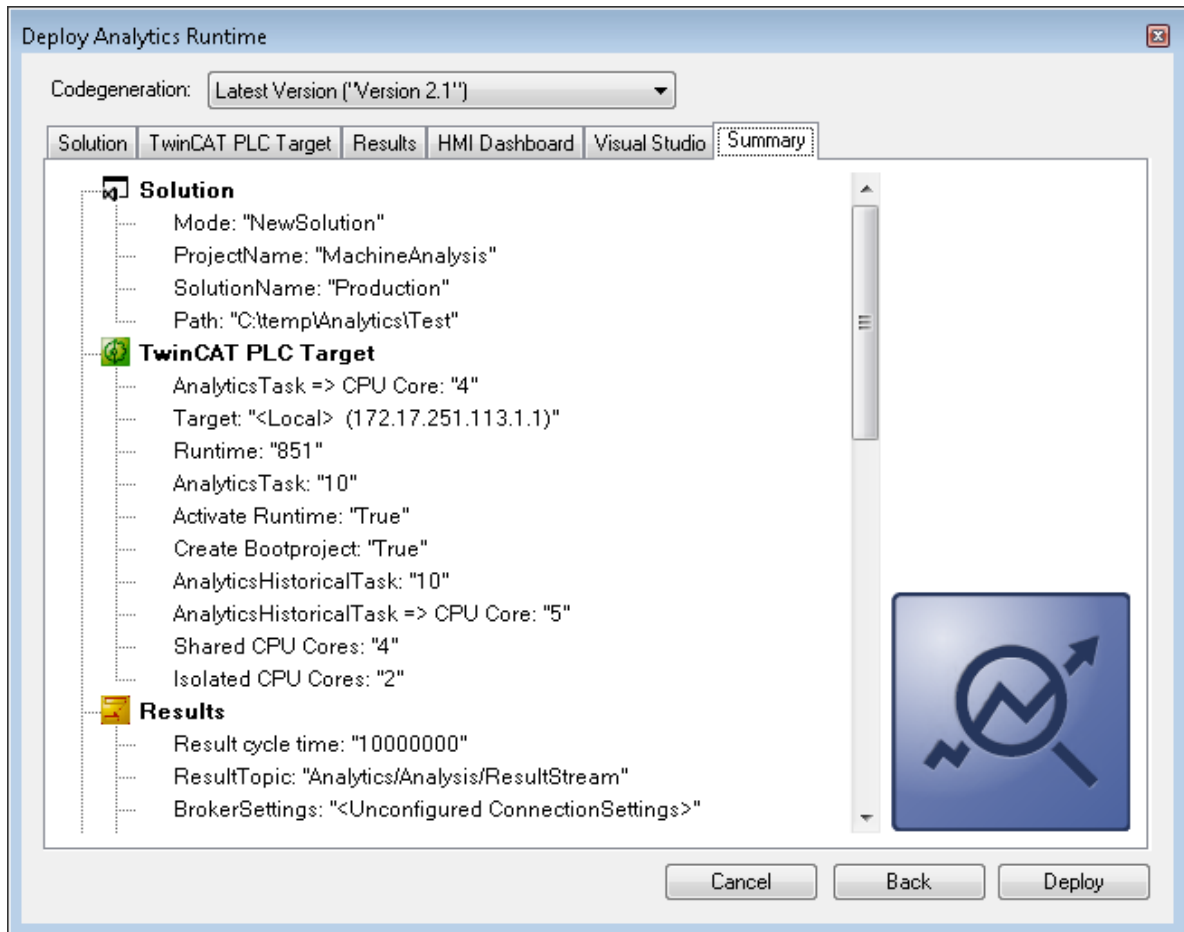
- You can choose different options for your Analytics Dashboard, such as a start page with a map, layouts, sorting algorithms, custom colors and logos. If you select multiple languages for the Analytics Controls, a language switching menu will also be generated.



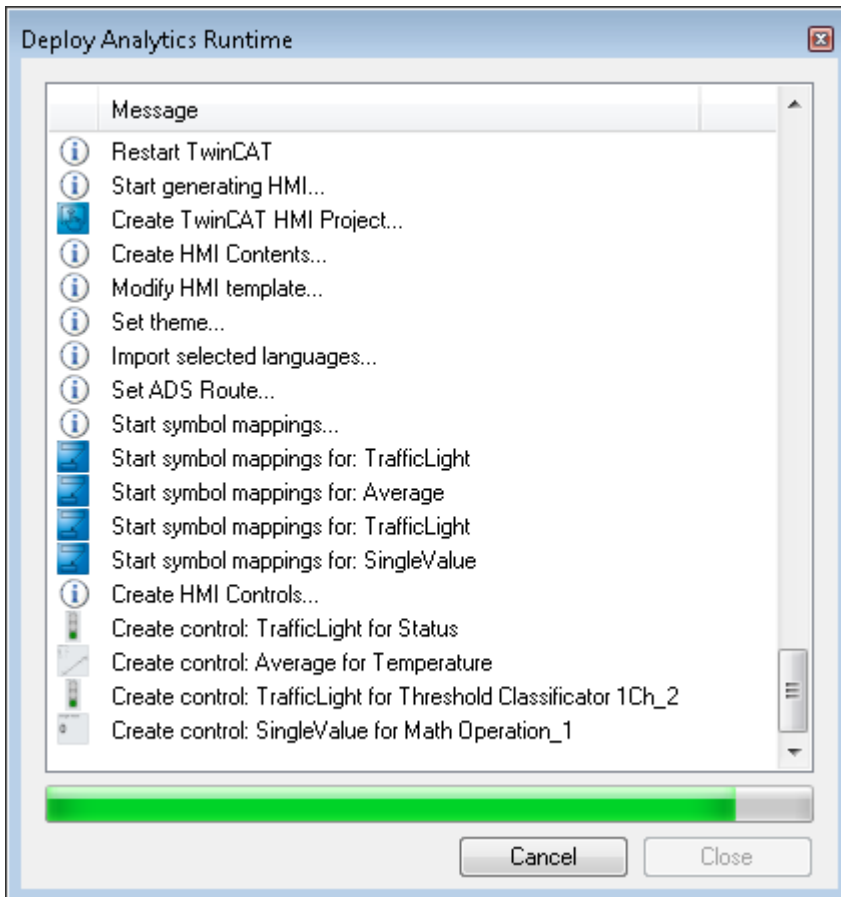
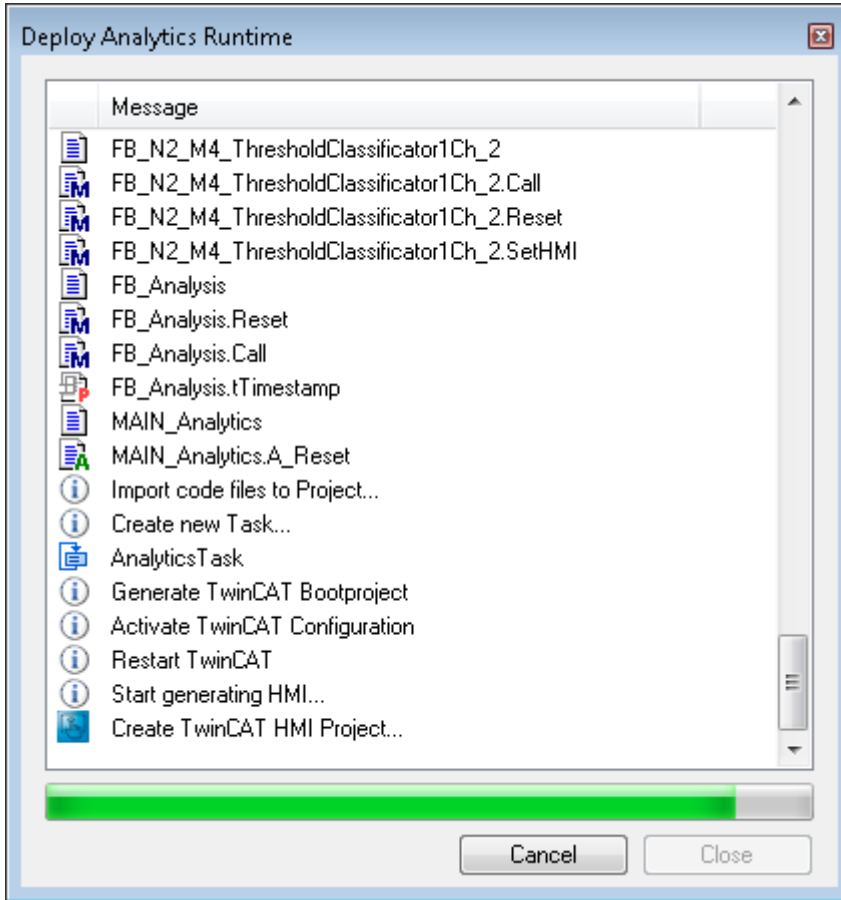
5. Select one of the installed versions of Visual Studio® and, whether the instance should start visibly or just be set up and activated in the background.



⇒ At last you can find an overview.

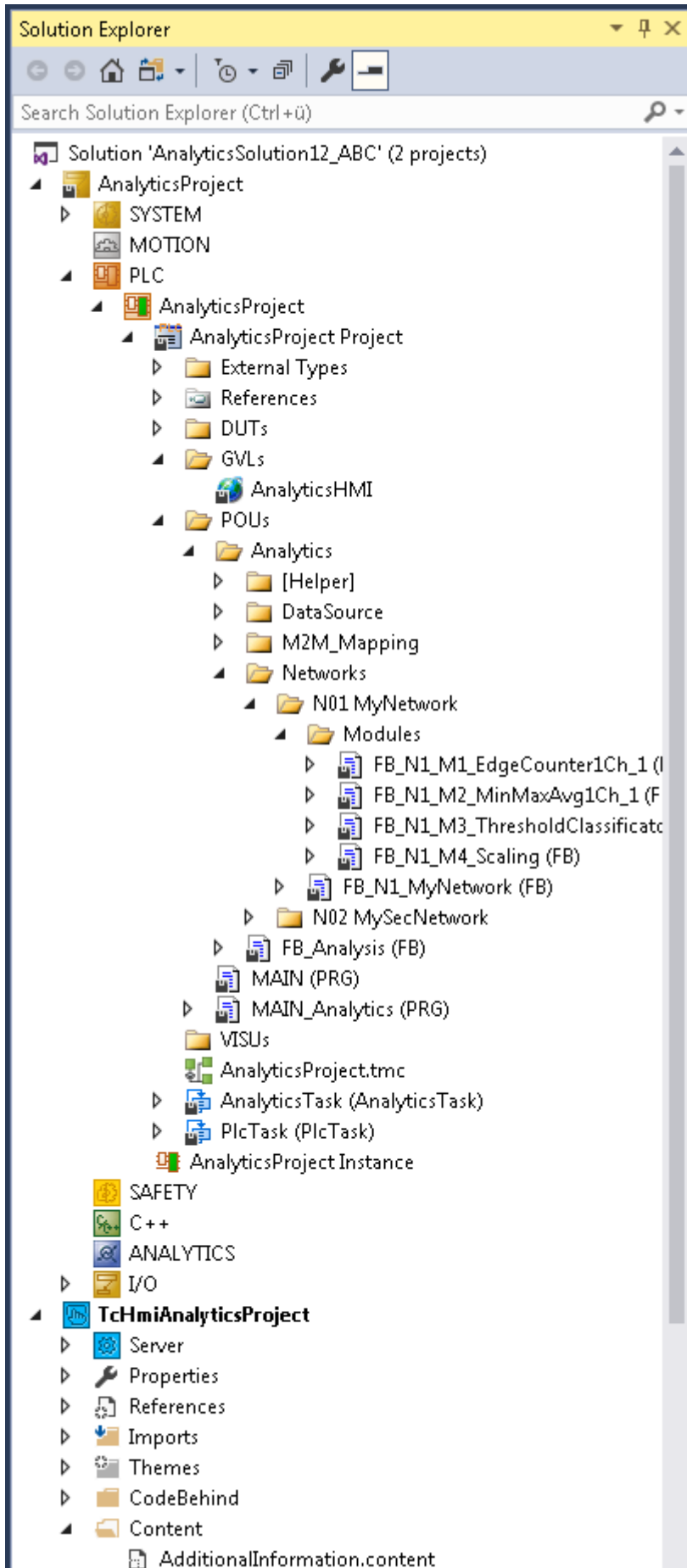


- Now you can click the **Deploy** button to start the generation process. The PLC project and the HMI dashboard are now generated.





⇒ After the "Deploy Runtime succeeded" message, you will find a new Visual Studio®/XAE shell instance on your desktop. The new Solution and both projects are created.



## 5 Technical introduction

The basic idea of the TwinCAT Analytics Storage Provider (ASP) is to have a gateway that largely frees the user from configuring a data sink, i.e. a storage or a database. The user does not need to set up his own table structure in a database. He only has to configure which of the supported data sinks he wants to use for storing his data.

### Service Management

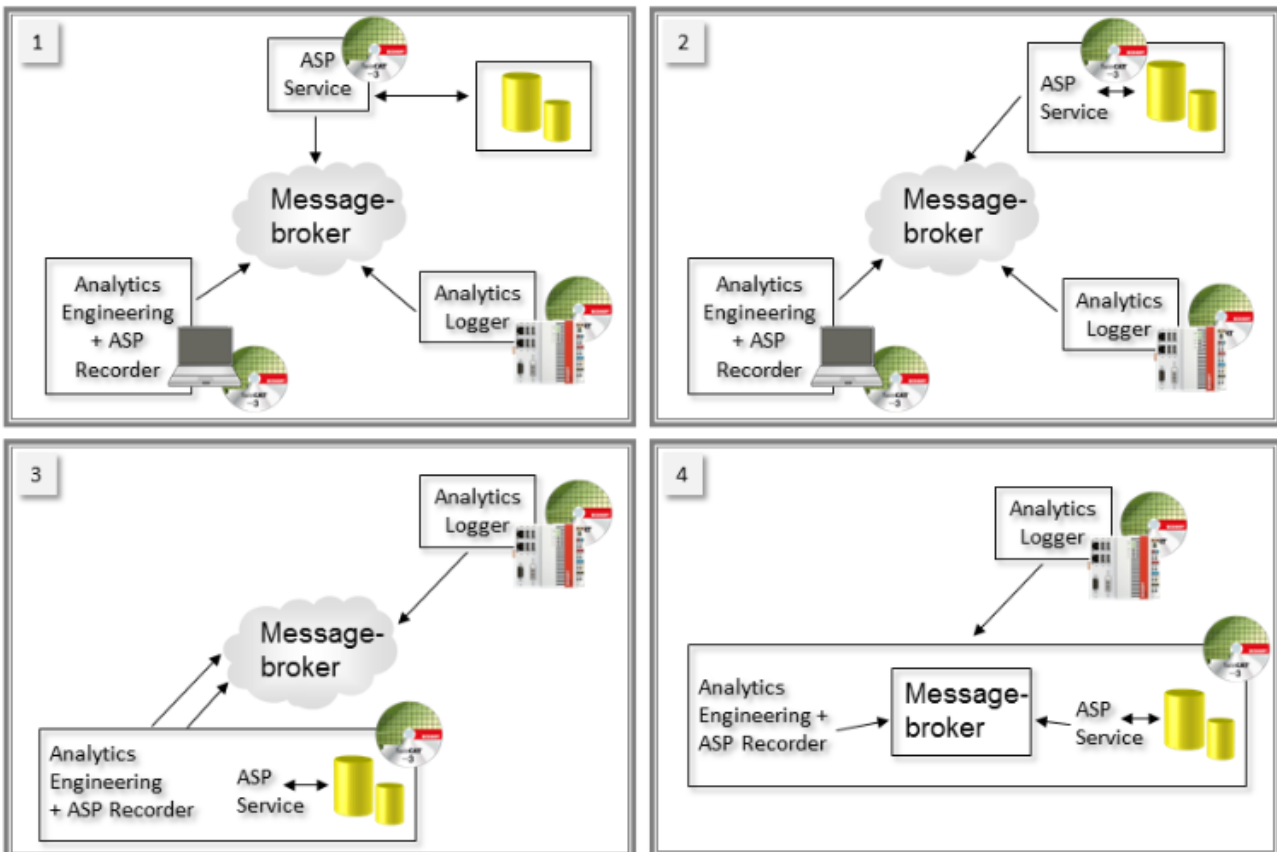
The Analytics Storage Provider service can run anywhere on the network. It is implemented as a Windows service. The service can run on hardware devices, such as industrial PCs or embedded PCs in the local network, and also on virtual machines in the same network, or in a cloud system, for example.

### Data Management

The Storage Provider works with the binary format of TwinCAT Analytics. This allows it to receive and store streams from an MQTT message broker and to create and send new streams itself. The user only needs the recorder, which is integrated with the TwinCAT Analytics Workbench or the service tool in his own engineering system. The variables themselves are displayed in the TwinCAT Target Browser. For the Analytics binary format, they are divided into live and historical data. Live data can be used as input to the Analytics Storage Provider. Historical data are the values from the database/storage provided by the Storage Provider.

### Topologies

The many degrees of freedom offered by IoT technologies enable different topologies. The following picture shows the most important constellations.



1. Each SW package runs on its own HW device or virtual machine.
2. The Analytics Storage Provider Windows service runs on the same device as the database/storage.
3. Analytics Engineering, Analytics Storage Provider, and database or storage are on the same device. Only the Message Broker and Analytics Logger (data source) run on other devices.

4. In this topology view, only the Analytics Logger runs on its own PC. This may be the case in a machine application. All other tools in the Analytics tool chain reside on one device, including the MQTT message broker.

**Topologies with additional ASP clients**

Currently, two additional clients are available from the Analytics Storage Provider perspective. A command line based client that allows execution from almost any application. And a PLC library that can also be used to influence the actions of the storage provider.

## 6 Configuration

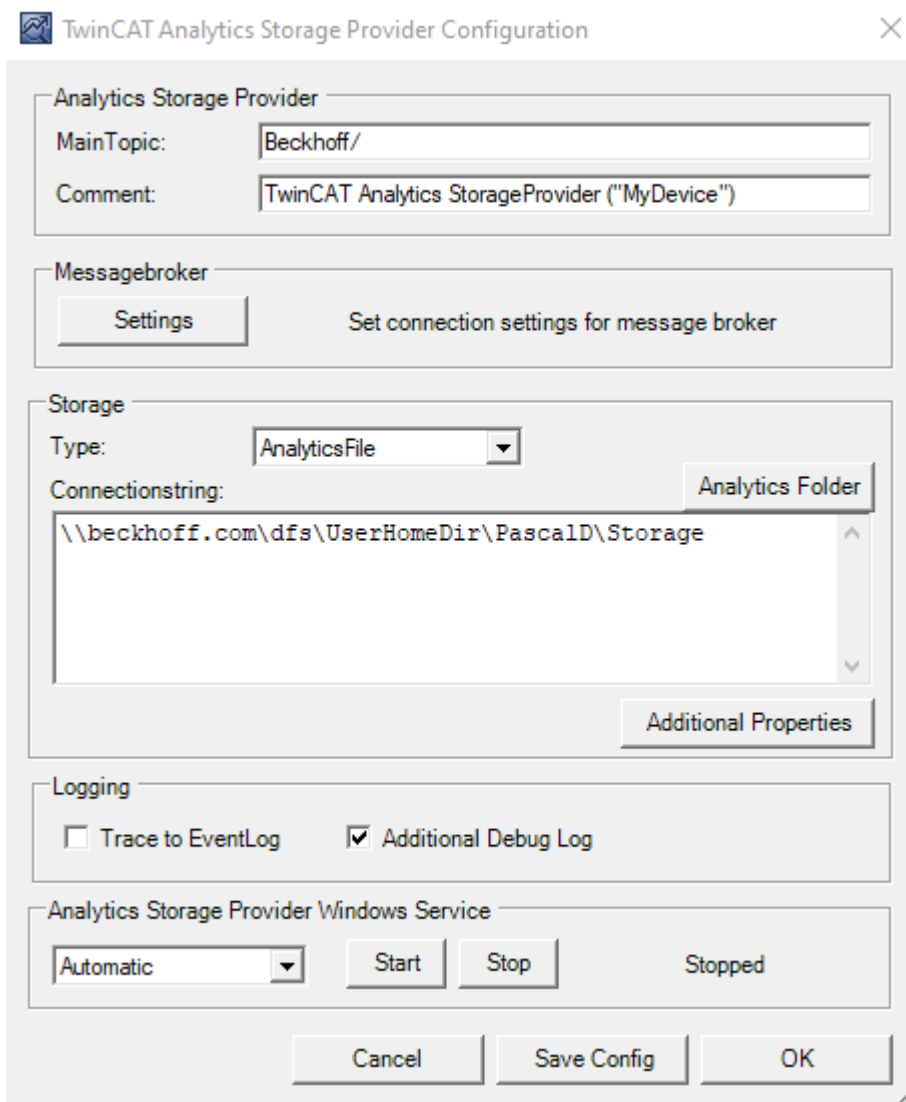
The configuration of the Analytics Storage Provider is divided into two main parts. The user has to configure the Service of the Storage Provider and variables by a Recorder. You will also find the supported databases and stores in this chapter.

### 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:

Test	Result
✓ Check broker host name [MyMessageBroker.cloudapp.net] IP:172.17.62.145	Success
✓ Check broker ip address [172.17.62.145] (Ping)	Success
✓ Check CA certificate 'MyCA.crt' ExpirationDate:03.12.2036 13:09:38	Success
✓ Connect client to broker	Success

✓ Connection settings: OK

- 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.

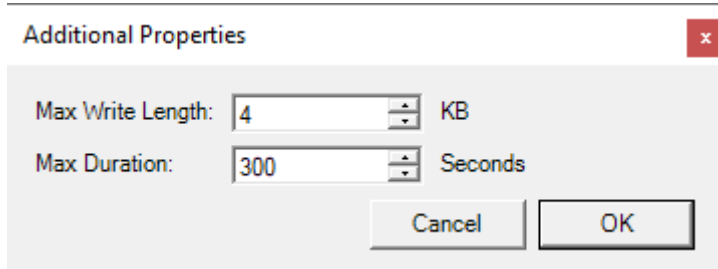
**Max Write Length:**

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



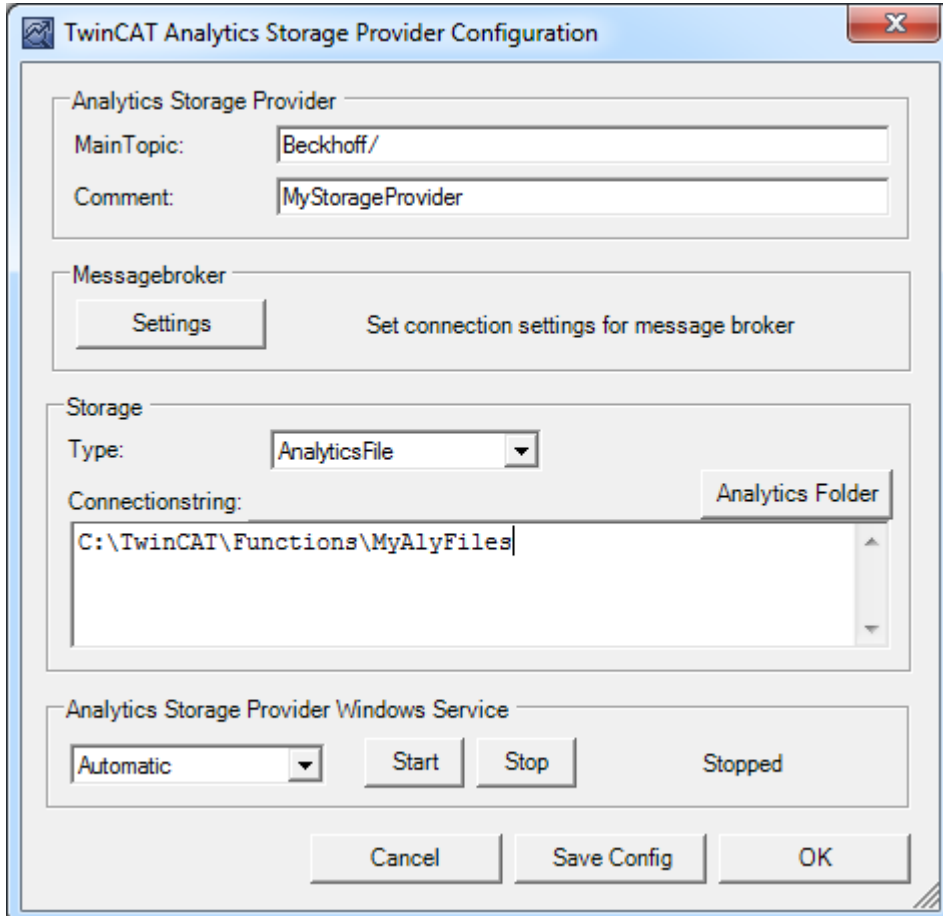
- Analytics Storage Provider Windows Service  
**Status:** View the status and default settings for starting the Windows service

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.

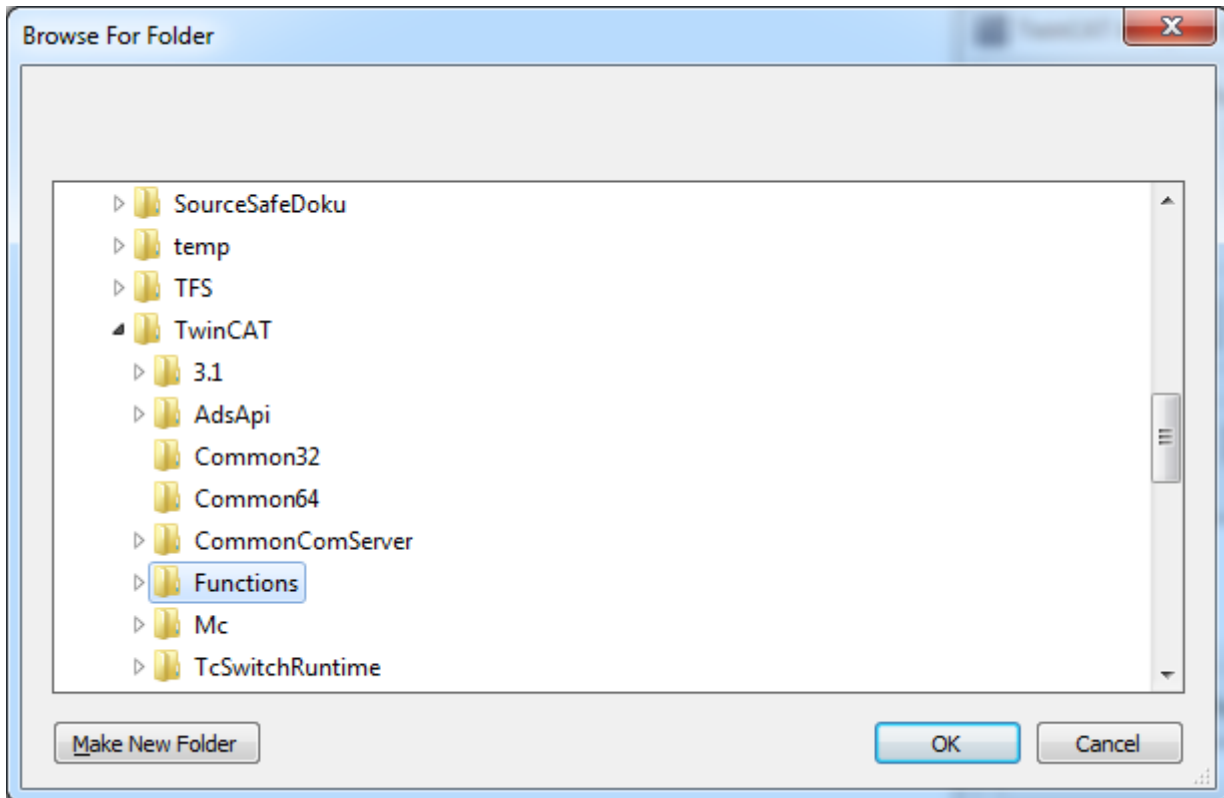
## 6.2 Databases/Stores

### 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.



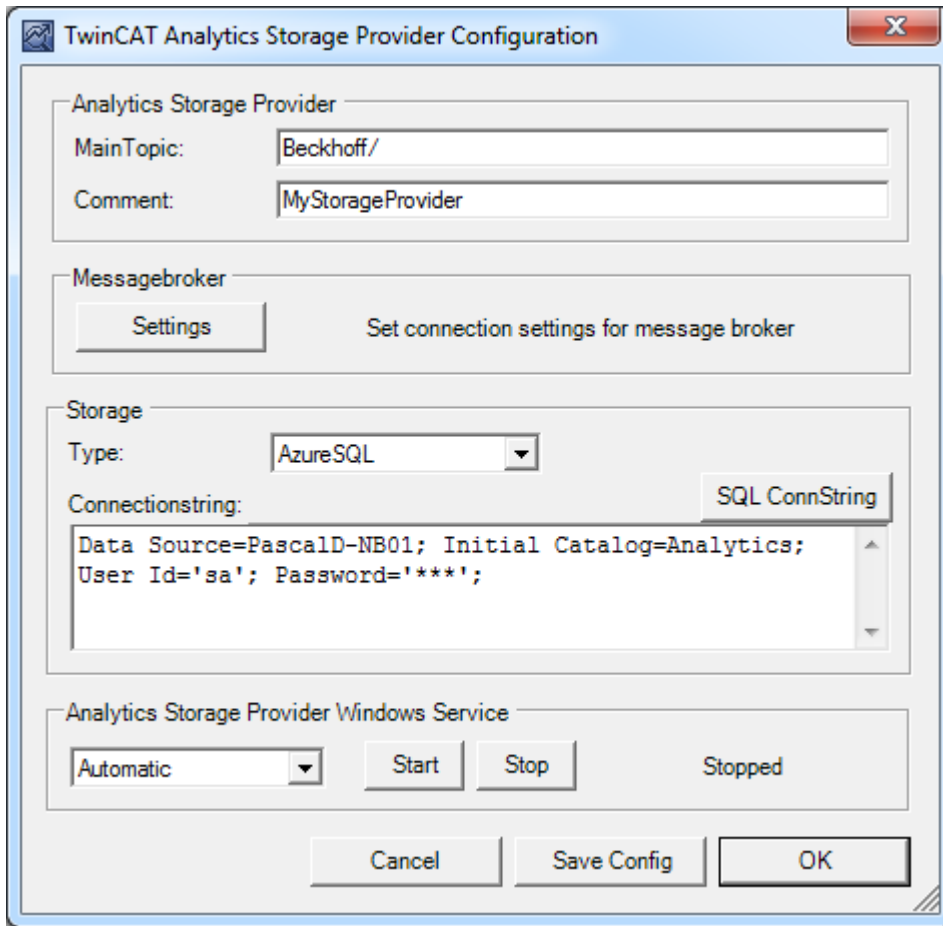
For the configuration it is enough that you choose your favorite folder on the local device where the Storage Provider is running.



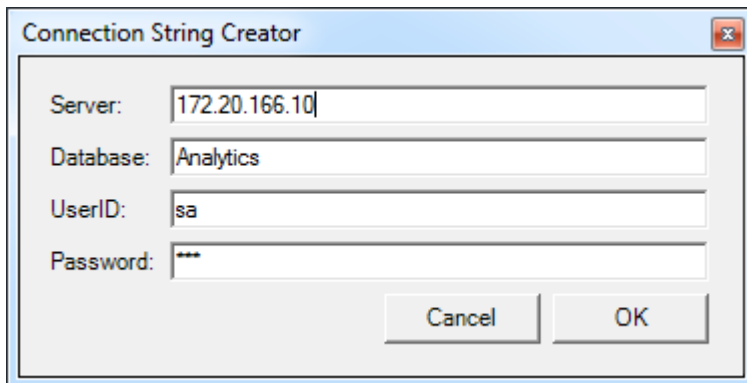
As confirmation you will see the used folder in the Connection String window.

## 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.

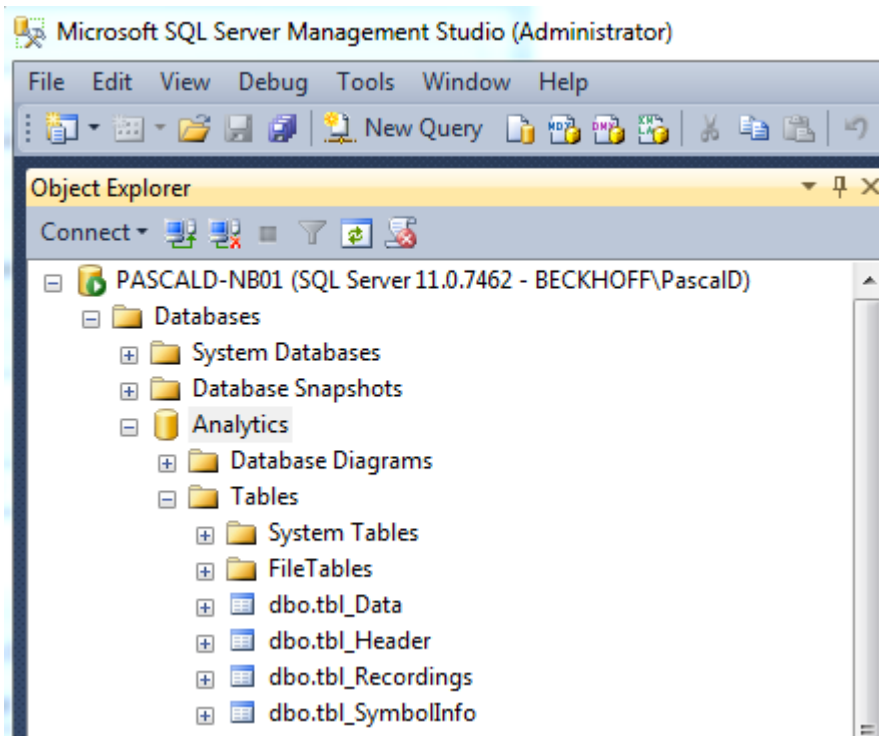


To make that simple you are able to click on the “SQL ConnString” button to open the input mask. There you can provide the configuration settings. Also of remote databases which are reachable by network connections.



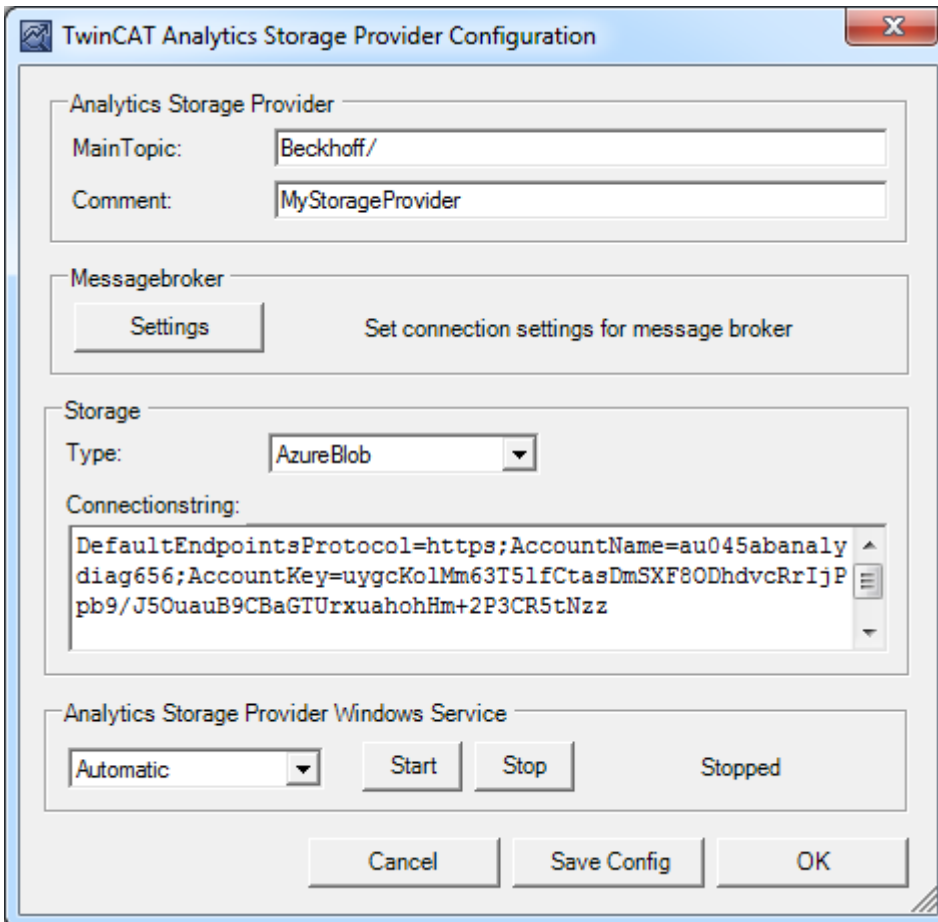
After starting the WinService and finishing the very first configuration of the Recorder tool we start the communication to the database. In this moment the Storage Provider is creating the four necessary tables by himself. As example see the following screenshot of Microsoft SQL Server Management Studio.



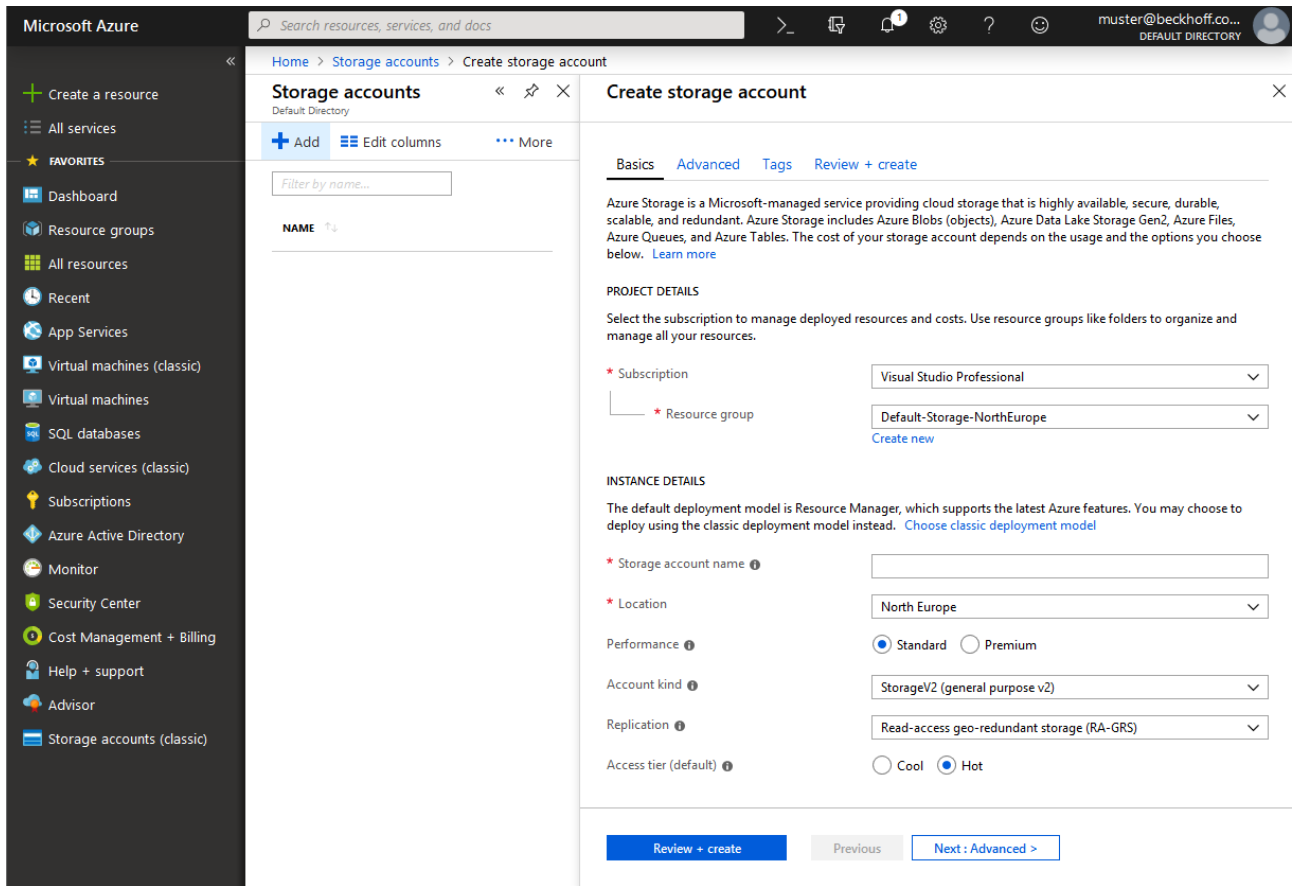


### 6.2.3 Microsoft Azure Blob

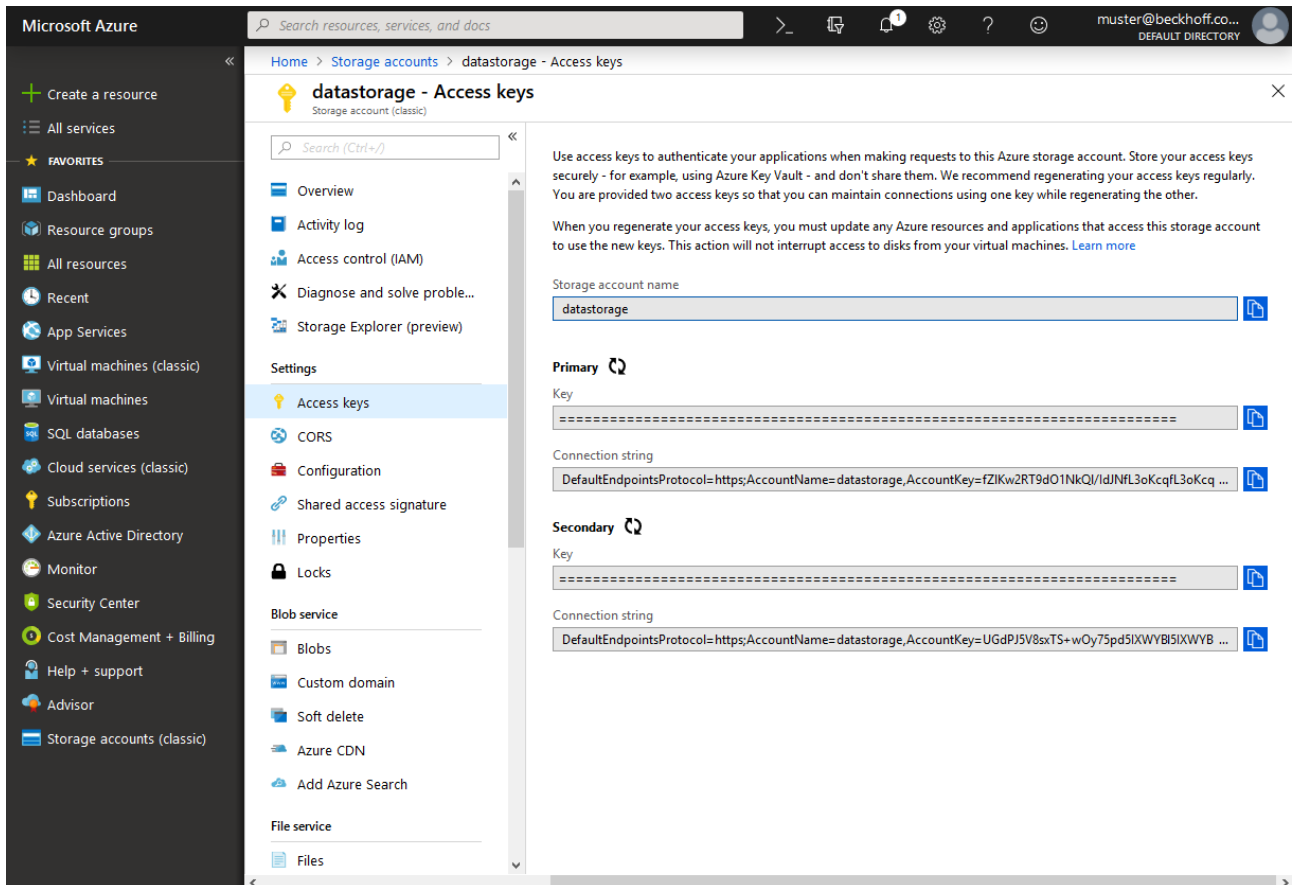
To use the Microsoft Azure Blob store you need to have a Microsoft Azure Cloud account. There you get also your individual connection string for the configuration of the TwinCAT Analytics Storage Provider.



You have to copy the connection string into the description field. In Azure itself, the storage is to be generated. Choose "Storage accounts (classic)".



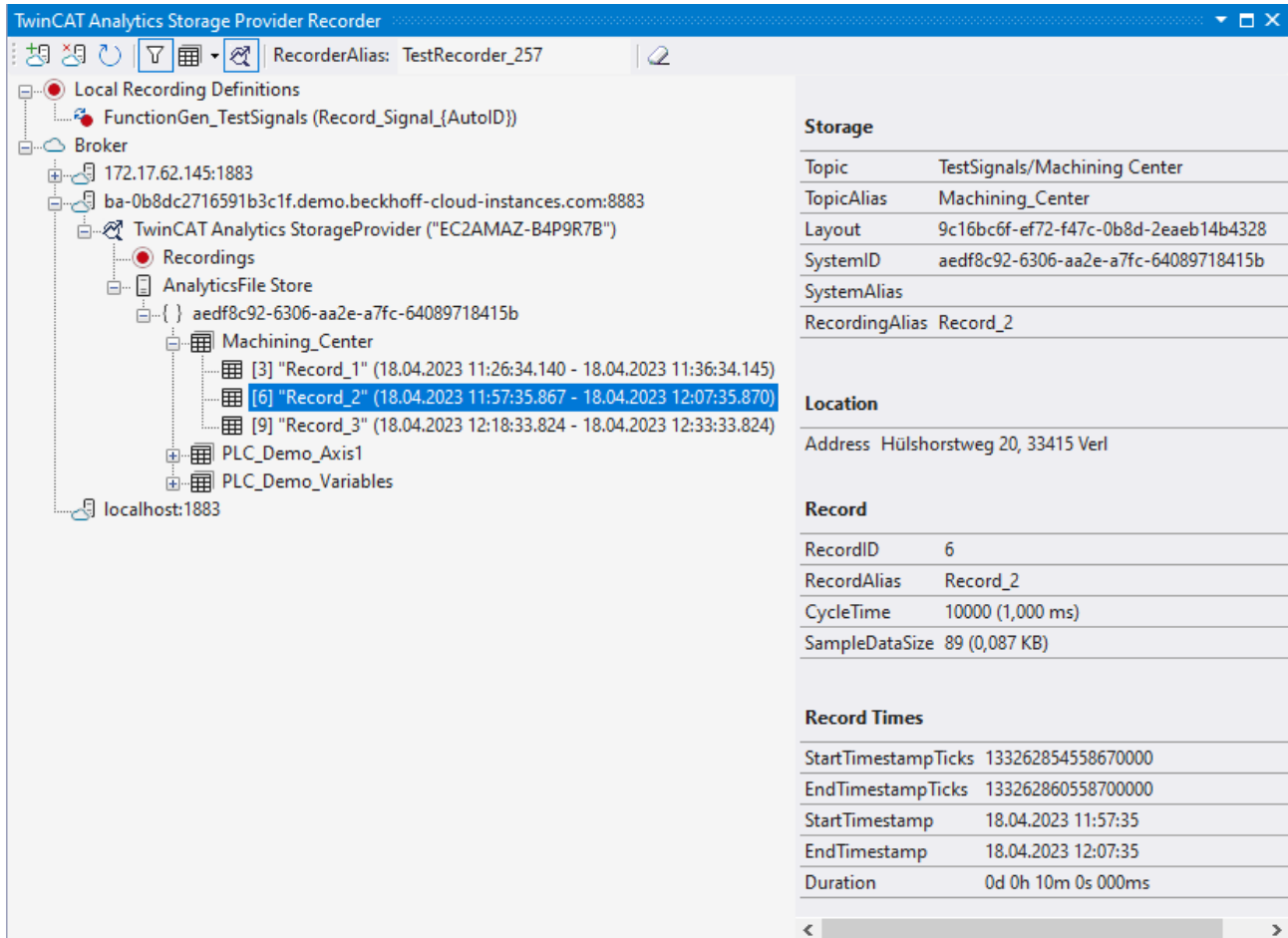
After generating the storage you will find under “Access keys” the secondary connection string. This string is to be use in the Analytics Storage Provider configuration.



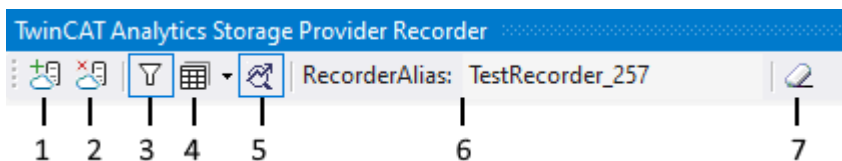
### 6.3 Recorder

The Analytics Storage Provider Recorder is part of the Analytics Engineering setups. Therefore, you can find the recorder in the installation of the TwinCAT Analytics Workbench and the TwinCAT Analytics Service Tool.

With the Storage Provider Recorder recording definitions can be created, started and managed. In addition, it is possible to manage the data memories of individual Analytics Storage Providers. All important properties of the found Analytics Storage Providers and historized data are clearly displayed.



#### Toolbar

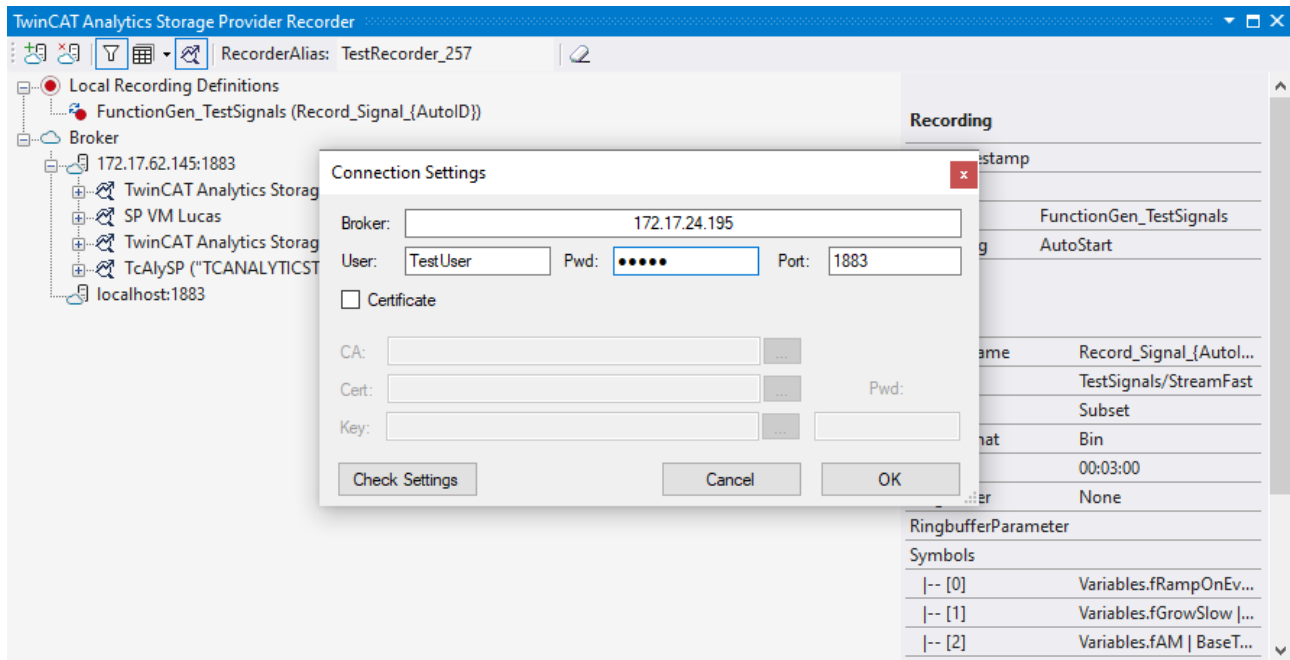


1	Add new broker
2	Remove selected broker
3	Display filters (All / Show my active recordings)
4	Select display type (Alias / Topics)
5	Show / Hide Offline Analytics Storage Provider
6	"RecorderAlias" - grouping name of recordings
7	Remove messages from error list

#### Recorder window setup

First assign a "RecorderAlias". This helps to group the started recordings and to find its self started ones again. The filter can also be used to display recordings started by others.

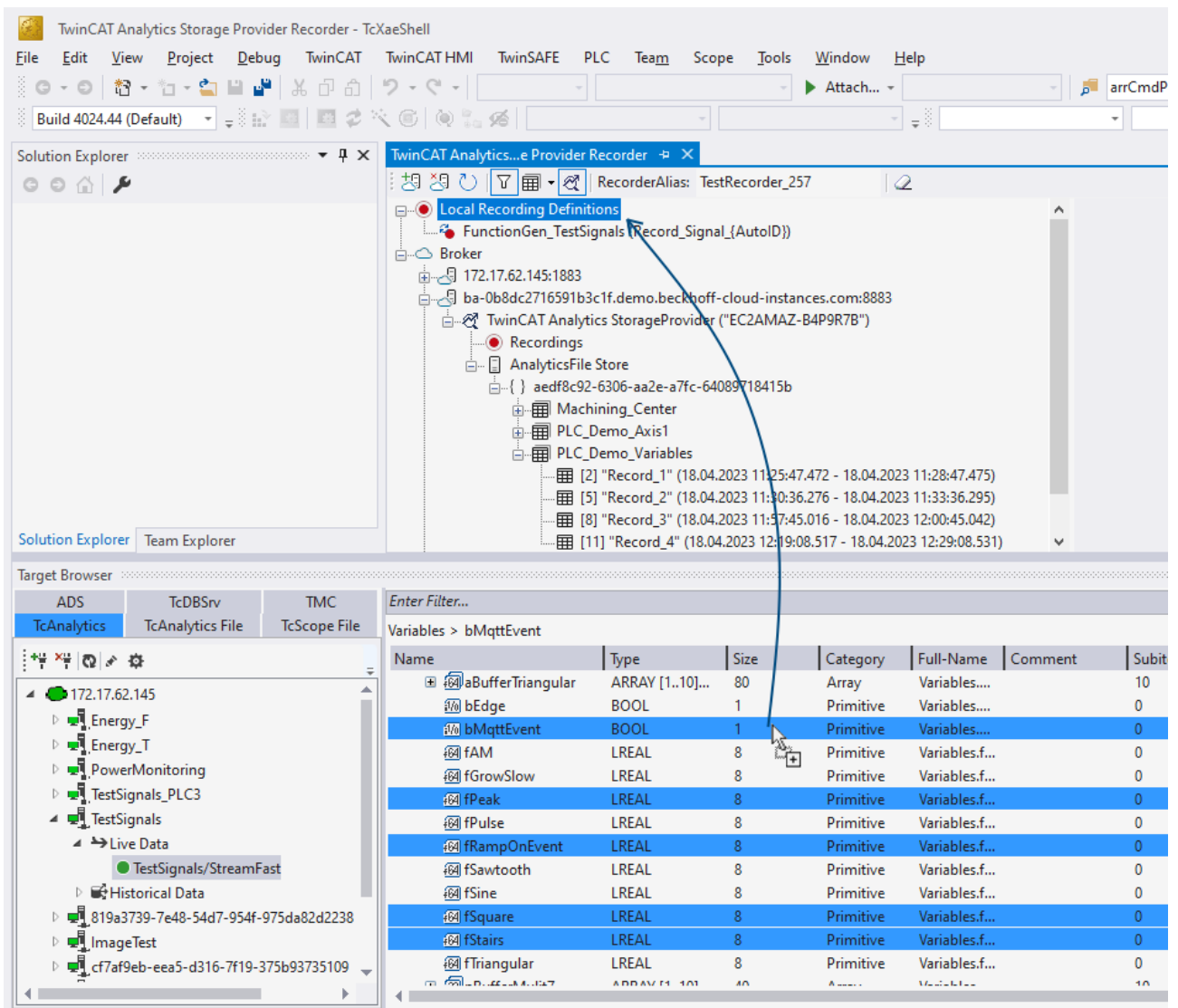
After that, one or more brokers can be set up. This is done via the already known input mask for MQTT connection properties.



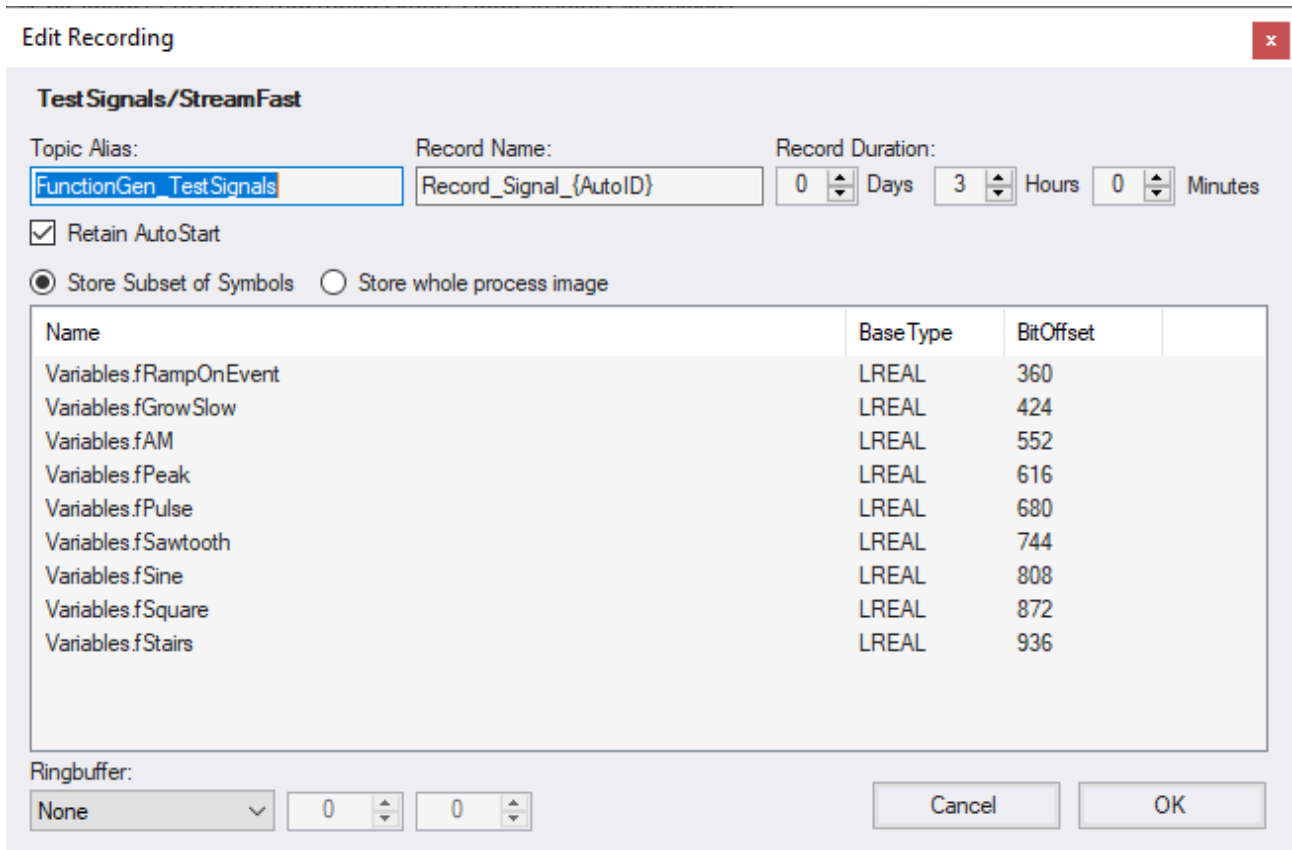
Once a connection to the broker could be established, all Analytics Storage Providers connected to it will be listed.

### Create recording definition

To configure the recording, select your target in the Target Browser. Click **Live Data** and select one or more variables by multiple selection and drag and drop them to the **Local Recording Definitions** node in the Recorder window.



In the recorder you can add the selected variables or the complete source process image of the variables.

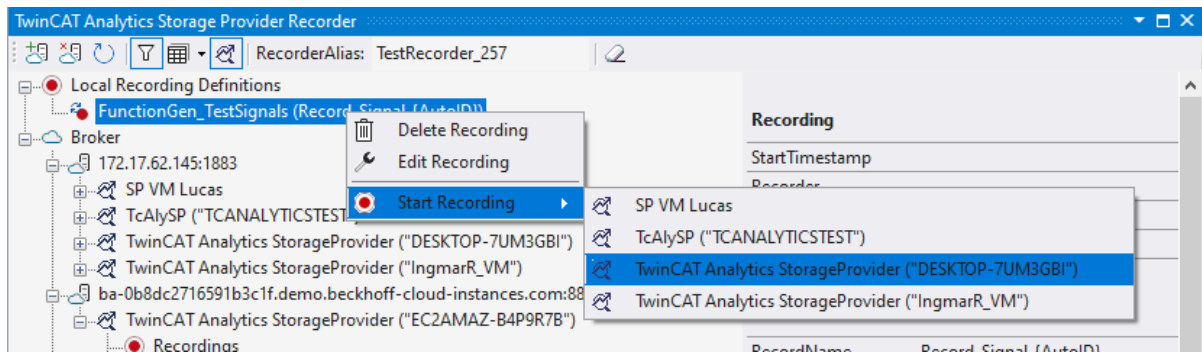


You can also configure recording names and a duration (otherwise the recording will run endlessly until it is stopped manually). A ring buffer can be set by storage or time.

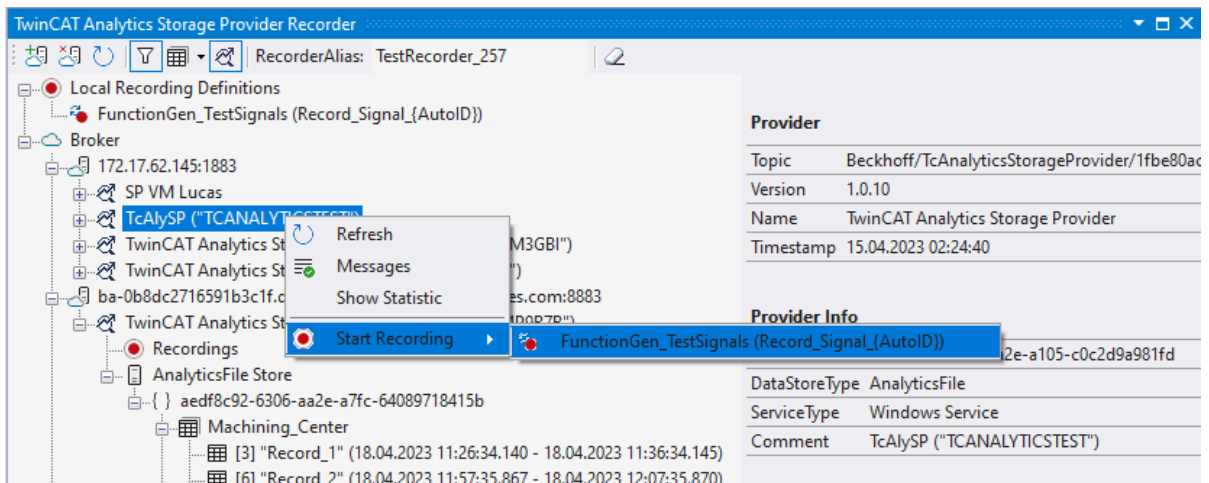
**Start recording**

There are three different ways to start a recording.

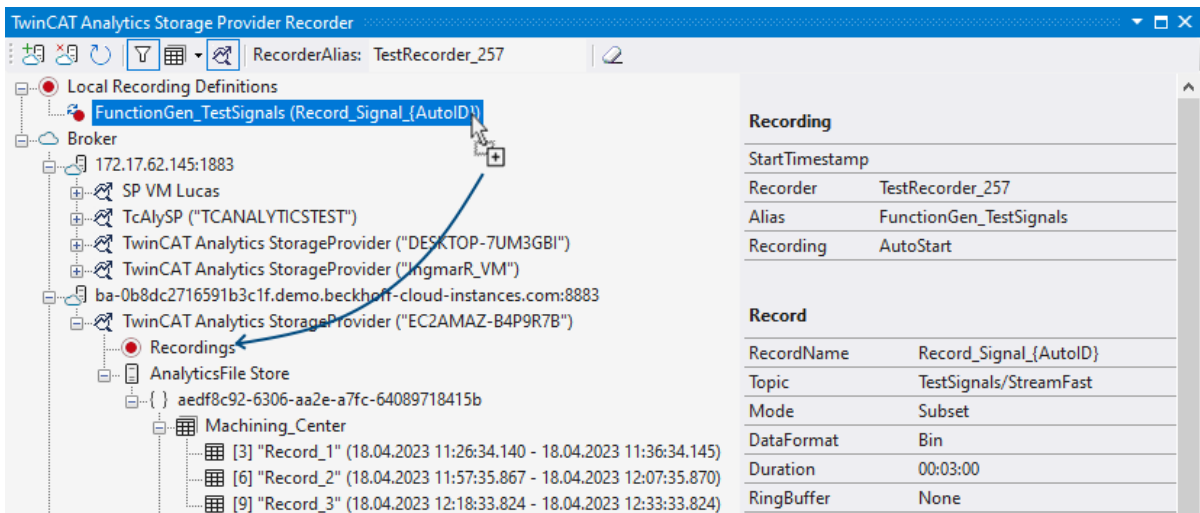
1. Via the context menu at a local recording definition node. Here you can select the desired Analytics Storage Provider, which should record the data. Only the storage providers that have access to the selected topic are displayed.



- From the context menu on the Analytics Storage Provider node. The desired recording definition can be selected here. Only the definitions that can also be processed by the Analytics Storage Provider are displayed.



- Drag and drop the local recording definition node. The desired recording definition can be started on the desired Analytics Storage Provider via drag-and-drop. If the Storage Provider is unable to execute the recording definition, an error message is displayed in the error list.



## 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.

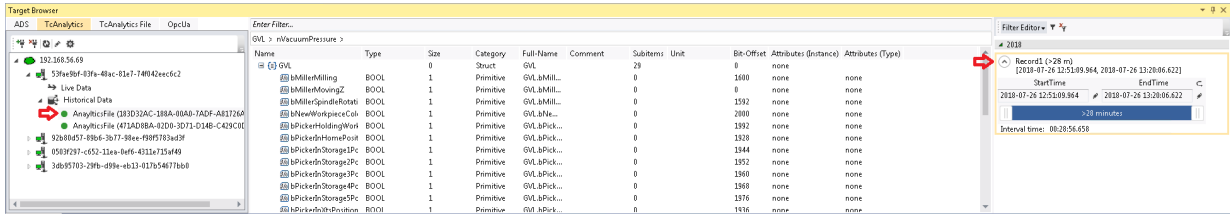
### Selection of data from the TwinCAT Target Browser

The historical data can be pulled directly from the Target Browser to an input of an analysis algorithm.

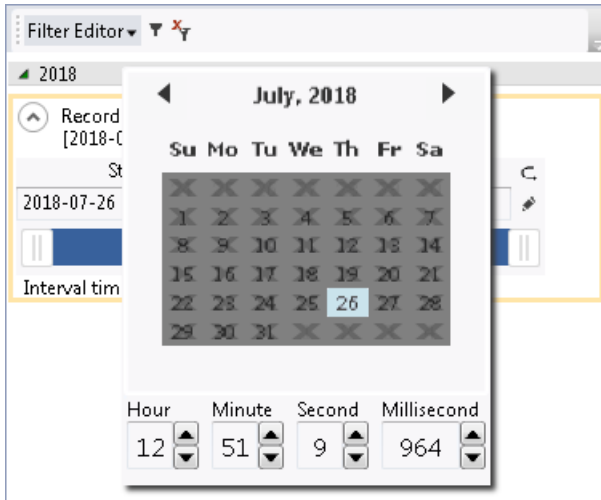
- First, you need to click **TcAnalytics** in the left corner of Target Browser. There you can see your configured broker, which lists live and historical data from your various devices. This should look like the following figure.



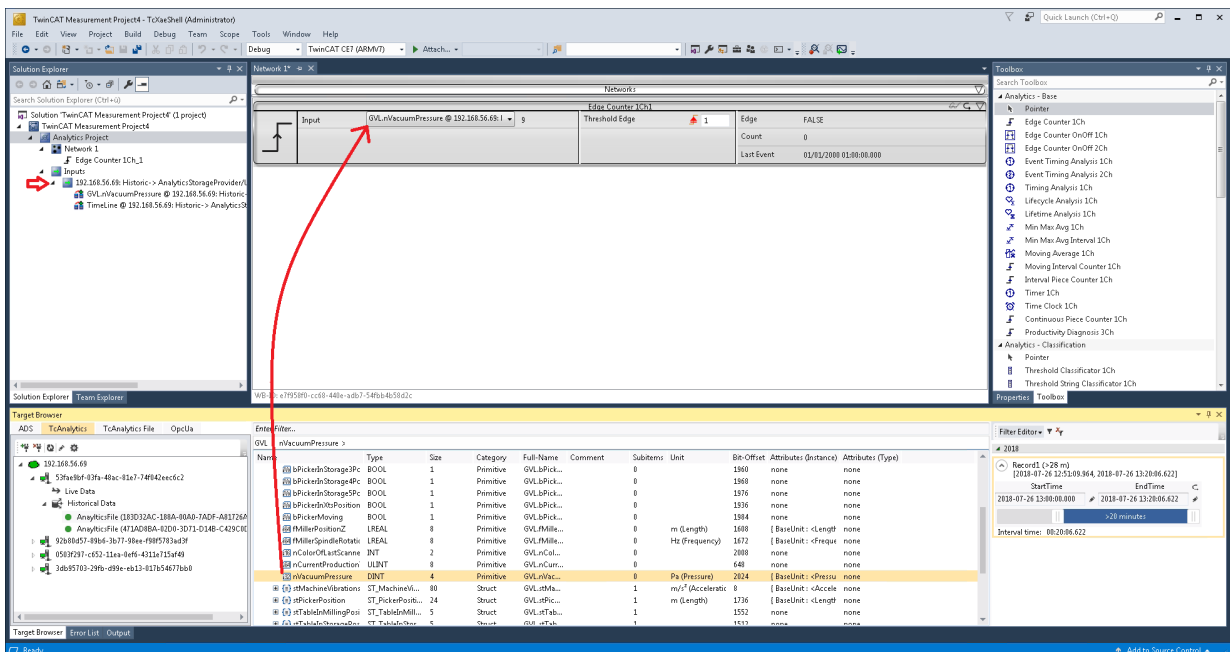
- Go to the historical stream you created and select the recording to be analyzed. All your records are listed in the **Record** window on the right. The last recording is selected by default.



- When you record live, the time range of the recording is updated every few seconds. The entire time range of a recording is used by default. You can also edit the start and end time to analyze your desired data area. This can be done with a slider, text fields or in a graphical calendar view. If you click on the symbol to the right of the text fields, the calendar view will be displayed.



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

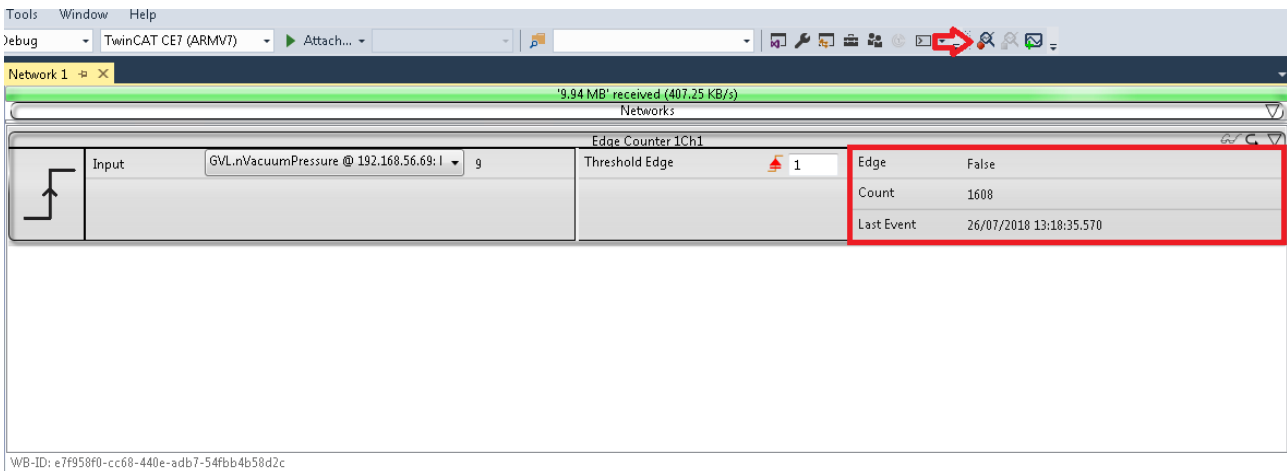


- ⇒ A new input source for your historical stream is then generated and can be displayed in the Solution Explorer of your Visual Studio®. First, the dragged symbol and a timestamp of the current device time are listed under this stream. Also new drawn symbols of this stream are listed there.



### 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.



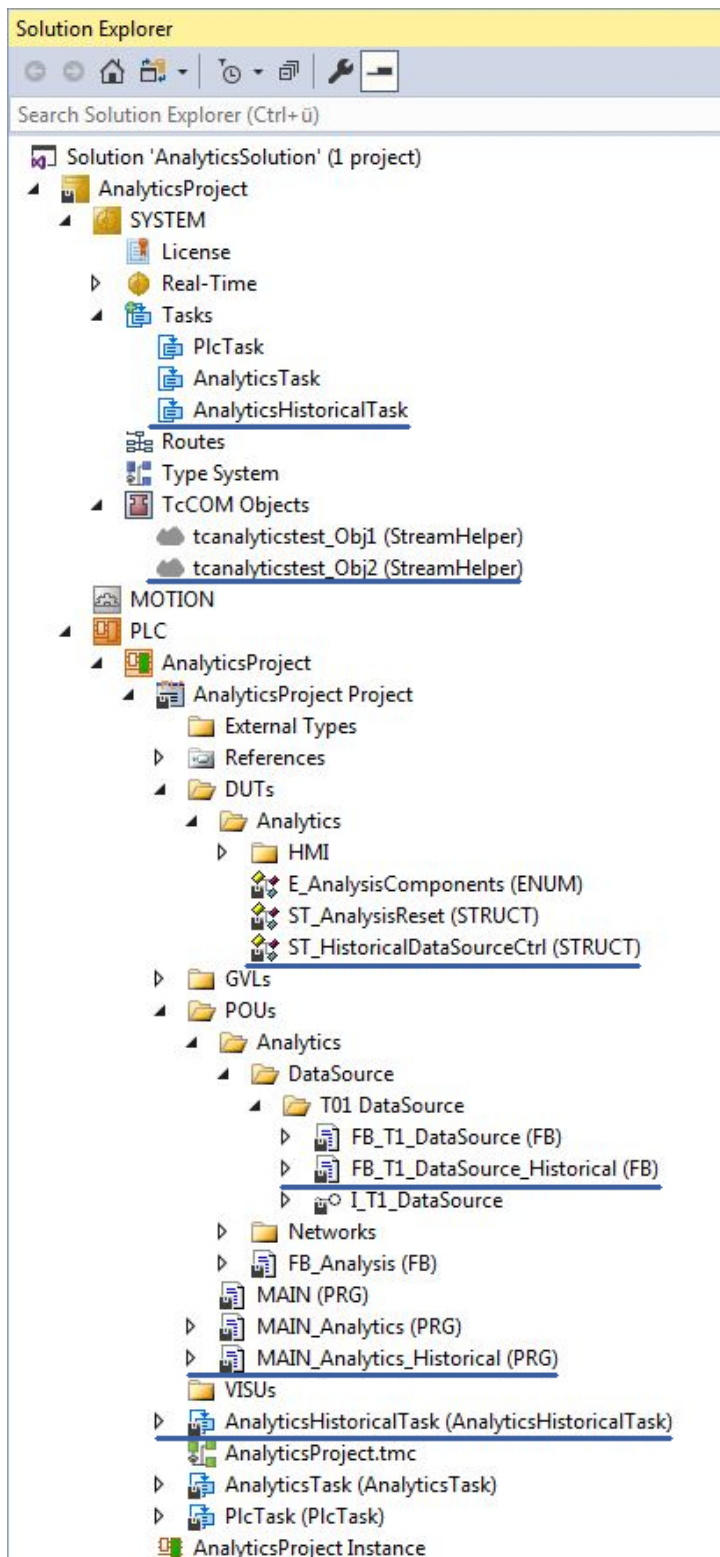
### Analysis of your historical data in your Analytics Runtime

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.

#### **i** 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 overruns due to excessively large packets must be taken into account.

Main differences of the folder structure in the created PLC project:



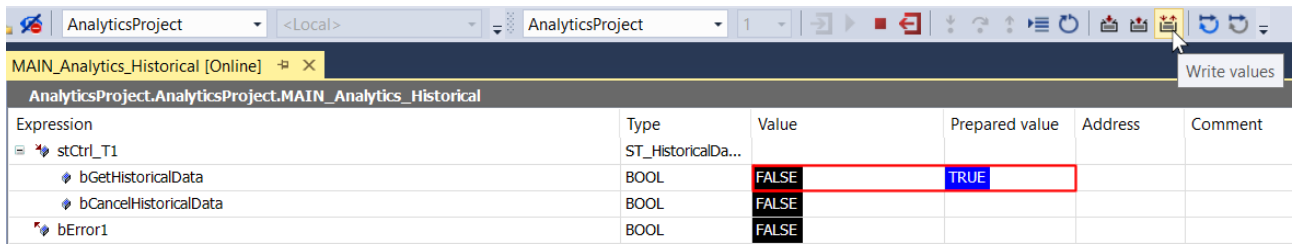
### NOTICE

#### Implementation of the logic in your TwinCAT HMI

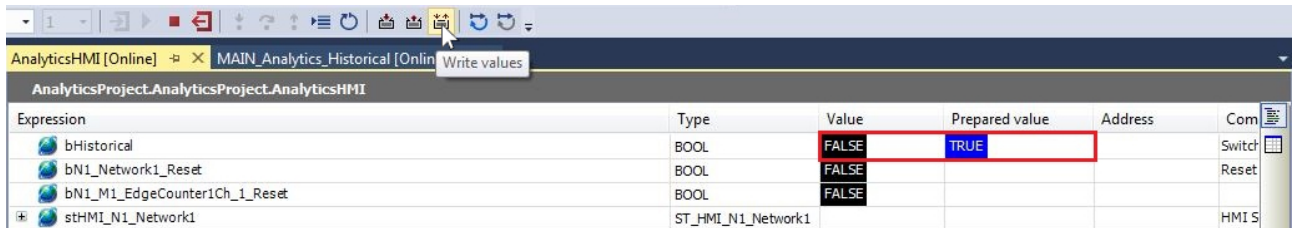
The preparation and writing of values in your PLC are for testing purposes. It is recommended to implement this and other logic in the PLC code with interactions from your TwinCAT HMI application if required.

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:



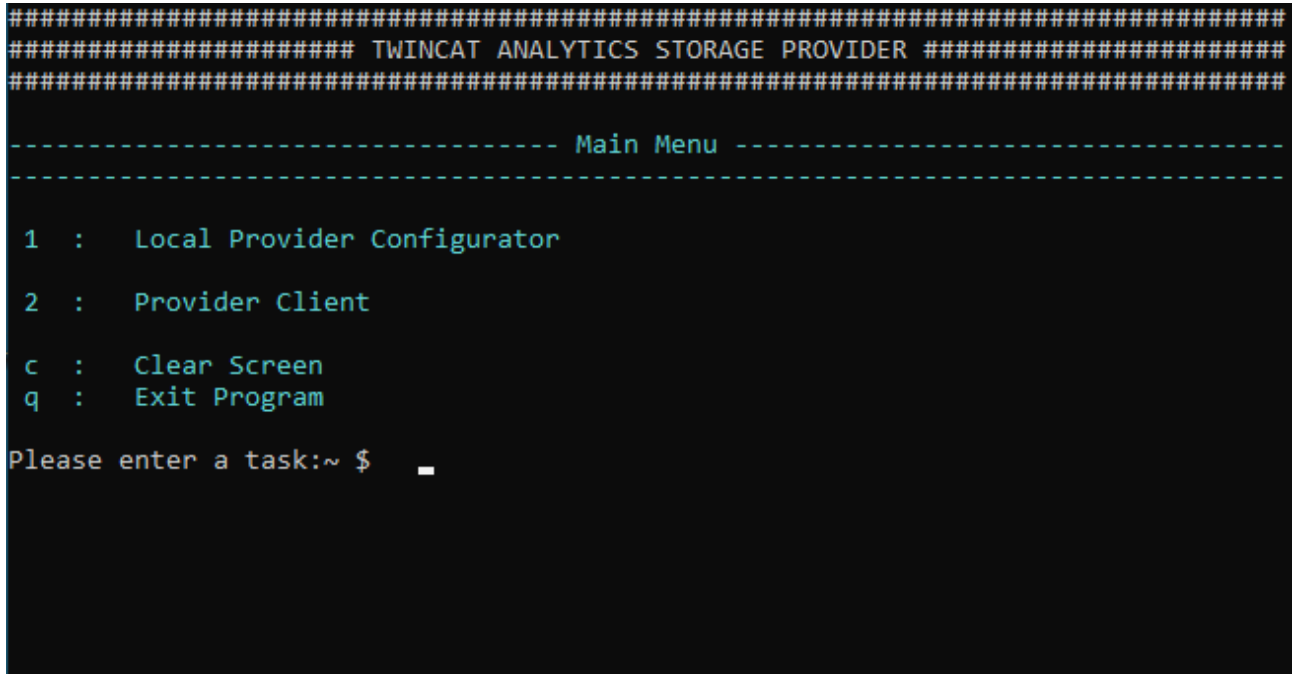
To switch between live and historical data results for your HMI dashboard, you can set the **bHistorical** symbol in the **AnalyticsHMI** GVL. With this option, you do not need any additional controls to display historical data (of course, you can also use your own controls for historical data). The 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.



## 6.5 Console Configurator/Client

In addition to the graphical configurator and the recorder, the Analytics Storage Provider can also be operated via a console. This means that configuration and access to a Storage Provider can also be performed under TwinCAT/BSD in addition to Windows. In addition, the console application can be used to generate Batch files for control [▶ 64] of the Analytics Storage Provider.

After launching the console client, there are four options to choose from:



1	Opens the console <u>Configurator</u> [▶ 60] for the local Analytics Storage Provider
2	Opens the Analytics Storage Provider <u>Client</u> [▶ 61]
c	Clears the console configurator/client history
q	Closes the console configurator/client

By entering one of the identifiers and confirming with the **[Enter]** key, the corresponding function is executed.

## 6.5.1 Configurator

In this menu, the local Analytics Storage Provider can be configured. The following additional inputs are available for this purpose:

<b>1</b>	Starts a dialog for configuring the local Analytics Storage Provider
<b>2</b>	Outputs the configuration of the local Analytics Storage Provider
<b>10</b>	Starts the Analytics Storage Provider with the local configuration
<b>11</b>	Stops the local Analytics Storage Provider
<b>12</b>	Displays the status of the Analytics Storage Provider
<b>b</b>	Switches back to the main menu

The configuration parameters are the same as in the graphical [configurator of the Analytics Storage Provider](#). To create the configuration, the configuration parameters are received one by one after entering "1":

```

##### TWINCAT ANALYTICS STORAGE PROVIDER #####
##### TWINCAT ANALYTICS STORAGE PROVIDER #####

----- Main Menu Local Provider Configurator -----
-----
1 : Create Local Provider Config
2 : Read Local Provider Config

10 : Start Local Provider Service
11 : Stop Local Provider Service
12 : Status Local Provider Service

b : Back to Start
c : Clear Screen
q : Exit Program

Please enter a task:~ $ 1

  Analytics Storage Provider Settings:
Please enter Provider Main Topic:~ $ Beckhoff/
Please enter Provider Comment:~ $ TwinCAT Analytics StorageProvider (Test)
Trace to EventLog (y/n):~ $ y

Additional Debug Log (y/n):~ $ y

  Messagebroker Settings:
Please enter Broker hostname/IP:~ $ localhost
Please enter Broker Port:~ $ 1883
Please enter Username:~ $ TestUser
Please enter Password:~ $ *****
Use certificates for connect? (y/n):~ $ n

  Storage Settings:
(1) - ANALYTICSFILE
(2) - MSSQL / AZURESQL
(3) - AZUREBLOB
Please choose id from storage type:~ $ 1

Please enter Storage path:~ $ C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\Storage
Please enter Max Write Length in bytes:~ $ 2048
Please enter Max Duration in seconds:~ $ 120
  Configuration successfully saved!

----- Main Menu Local Provider Configurator -----
-----
1 : Create Local Provider Config
2 : Read Local Provider Config

10 : Start Local Provider Service
11 : Stop Local Provider Service
12 : Status Local Provider Service

b : Back to Start
c : Clear Screen
q : Exit Program

Please enter a task:~ $

```

When selecting the storage type, **AnalyticsFile** is the default storage type in case of a wrong entry. By pressing the [ESC] key, the dialog can be aborted at any time and you can return to the configuration menu.

## 6.5.2 Client

In the Analytics Storage Provider Client it is possible to connect to an Analytics Storage Provider. This does not necessarily have to run locally on the device, but can also be addressed via an external MQTT broker. Thus, it is not mandatory to configure an Analytics Storage Provider locally to use the client. Recordings can then be started and stopped on a connected Analytics Storage Provider, as well as historical data streams to a configurable MQTT topic.

The following inputs are available for this:

1	Establishes a connection with the MQTT broker from the local configuration and selects the configured Analytics Storage Provider
2	Starts a dialog to connect to an MQTT broker
3	Closes the connection to the current MQTT broker
10	Provides the analytics storage providers available under the MQTT broker for selection
20	Reads in a configuration file from an <u>Analytics Storage Provider Recorder</u> , with the recordings configured in it
31	Starts a recording
32	Stops a recording based on the alias and the MQTT topic
33	Starts a historical data stream
34	Stops a historical data stream based on the result MQTT topic
35	Updates streaming parameters of a running historical data stream
36	Checks whether a recording is active
40	Stops all active recordings
41	Stops all historical data streams

To use the Analytics Storage Provider Client, a connection to an MQTT broker must first be established. Entering "2" starts a dialog in which the already configured MQTT brokers are presented for selection. There a new MQTT broker can be configured and connected by entering "0". Then, by entering "10" in the client main menu, an Analytics Storage Provider can be selected, which is available under the MQTT broker. Alternatively, by entering "1" in the client main menu, a connection to the MQTT broker and Analytics Storage Provider can be established directly from the local configuration file.

After the connection is successfully established, information about the connected MQTT broker and the selected Analytics Storage Provider is displayed in the prompter display before the prompt:

```
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter a task:~ $
```

(MQTT broker: Analytics Storage Provider)

To start recordings, a dialog is started by entering "31". It is possible to start a recording that has already been created, provided that recordings have already been configured or read in via a recorder configuration file. In addition to the listed recordings, a new recording can also be configured and started by entering "0":

```

RecorderAlias:Console ASP Client ("MARCT-NB01") - [ffca662c-0000-0000-0000-80d643e8d241] ← 1
----- Main Menu Provider Client -----
1 : Use Local Provider
2 : Connect
3 : Disconnect

10 : Select Provider

20 : Load Config (RecorderSettings)

31 : Start Recording
32 : Stop Recording
33 : Start Historical Stream
34 : Stop Historical Stream
35 : Set Historical Stream Parameter
36 : Is Recording Active

40 : Cancel All Recordings
41 : Cancel All Historical Streams

b : Back to Start
c : Clear Screen
q : Exit Program

[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter a task:~ $ 31
Recordings:
(0) - [Create new recording]
(1) - "TcBSD_ASP_Recording" (ASP_Record)
(2) - "TcBSD_ASP_Alias" (TcBSD_Record)
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please choose id from record config:~ $ 0

[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter topic:~ $ TestSignals/StreamFast
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter recording alias:~ $ ASP_Recording
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter recordname:~ $ AnalyticsSP_Record
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter duration in minutes:~ $ 120
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter ringbuffer [None|TimeBased|DataBased]:~ $ None
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter dataFormat [Bin|Json]:~ $ Bin
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter recording mode [All|Subset]:~ $ All
Recording command "ASP_Recording" send to Provider.
Recording "ASP_Recording" is running.

```

The configuration parameters correspond to the known parameters from the graphical [Analytics Storage Provider Recorder](#). The default values can be deleted if necessary and replaced by individual entries. With the **recording mode** after the input "Subset" a subset of the symbolism can be defined by the recording data. Immediately after configuration, a command to start recording is sent to the connected Analytics Storage Provider. Running recordings can be stopped by entering "32". They are referenced by the MQTT topic from which the data comes and by the recording alias. If a recording is to be stopped by another client, the corresponding **Recorder Guid** must also be specified. The **Recorder Guid** is displayed together with the **Recorder Alias** above the input options in the client main menu (red 1).

Recording configurations created in the console client are not persisted. So after closing the client, the list of recording configurations is no longer available. Therefore, reading recorder configuration files (enter "20" in the client main menu) can be very helpful. The configuration file of a recorder is stored on Windows systems under the path `C:\Users\***\AppData\Roaming\Beckhoff\TwinCAT Analytics Storage Provider` (replace \*\*\* with the corresponding user).

The historized data of the Analytics Storage Provider can be transmitted as a data stream to a definable result MQTT topic via the input "33". This also starts a dialog in which a previously configured data stream can be started. By entering "0", a new historical data stream can also be configured:

```

1 : Use Local Provider
2 : Connect
3 : Disconnect

10 : Select Provider

20 : Load Config (RecorderSettings)

31 : Start Recording
32 : Stop Recording
33 : Start Historical Stream
34 : Stop Historical Stream
35 : Set Historical Stream Parameter
36 : Is Recording Active

40 : Cancel All Recordings
41 : Cancel All Historical Streams

b : Back to Start
c : Clear Screen
q : Exit Program

[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter a task:~ $ 33
GetHistorical Cmds:
(0) - [Create new GetHistorical cmd]
(1) - "RecordID:29 | Topic:TestSignals/StreamFast"
(2) - "RecordID:39 | Topic:TestSignals/StreamFast"
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please choose id from cmd:~ $ 0

[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter systemID:~ $ cff7975b-b34d-43f7-755d-95cf135f50db
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter topic:~ $ TestSignals/StreamFast
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter symbol layout:~ $ 52a5066f-3c94-d853-f02b-bce62b4a6dea
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter record id:~ $ 1
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter start time in ns:~ $ 13331821883950000
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter end time in ns:~ $ 133318220040054000
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter maxSampleCnt default:~ $ 5000
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter user sampletime default:~ $ -1
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter outputFormat [Bin|Json]:~ $ Bin
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter result topic:~ $ TcBSD_AnalyticsSP_ResultTopic
[TcAnalyticsTest : ASP TcBSD (CX-5AC79C)] Please enter symbol mode [All|Subset]:~ $ All
GetHistoricalStream command for "TcBSD_AnalyticsSP_ResultTopic" send to Provider.

```

The parameters define a historized recording, whereby the parameter **result topic** defines the MQTT topic to which the data is to be streamed. After configuration, a command to start the historical stream is automatically sent to the Analytics Storage Provider.

By entering "35" in the client main menu, the parameters of an active historical stream can be adjusted. The historical stream is referenced by its result MQTT topic. The parameters can be used, for example, to adjust the speed or packet size of the data stream while it is running. Canceling a historical data stream is possible by entering "34" and specifying the result MQTT topic.

### 6.5.3 Batch files for control

The console client can be used to create batch files to control the Analytics Storage Provider. Some parameters are provided for this purpose:

-Help / -H / -?	Returns a description of all parameters
-----------------	---

Parameters for the configuration settings:



-CreateASPConfig	Create a new Analytics Storage Provider settings XML
-MainTopic <mainTopic>	Analytics Storage Provider Main Topic
-Comment <comment>	Analytics Storage Provider comment
-EventLogTrace <True False>	Trace to the event log
-DebugLog <True False>	Additional DebugLog
-StorageType <type>	Storage type ( <b>ANALYTICSFILE, AZURESQL, AZUREBLOB</b> )
-StorageConnString <connString>	Connection string or path to the memory
-TlsType <Tls1.0 Tls1.1 Tls1.2>	Tls type (for AzureBlob)
-MaxDuration <duration (sec)>	Maximum duration of a TAY file
-MaxWriteLen <writeLen (bytes)>	Maximum length of a data packet

Configuration parameters:

-LocalProvider	Use the connection settings of the locally installed Analytics Storage Provider
-ConfigFile <path>	Use all configurations from the configuration file of an Analytics Storage Provider Recorder window
-ProviderGuid <guid>	Provider of the Analytics Storage Provider to be used
-ConfigCmdID <id>	ID number of the preconfigured recording in the configuration file
-ConfigCmdAlias <alias>	Alias of the preconfigured recording in the configuration file

Connection parameters:

-Broker /-Host <hostname>	Host name or IP address of the broker used
-Port <port>	Broker port (default value: 1883)
-User <username>	Username for the connection
-Password / -Pwd <password>	Password for the connection
-CA <path>	Path to the CA certificate for the connection
-Cert <path>	Path to the certificate for the connection
-Key_Cert <path>	Path to the key file for the connection
-Key_Pwd <password>	Password for the key file for the connection

Function parameters:

-StartRecord	Sends a StartRecord command
-StopRecord	Sends a StopRecord command
-IsRecordingActive	Checks whether a recording is currently running
-GetHistorical	Sends a GetHistoricalData command
-StopHistorical	Sends a StopHistoricalData command
-UpdateHistorical	Sends a HistoricalUpdate command
-CancelAllRec	Sends a Cancel command to all active recordings
-CancelAllHist	Sends a Cancel command to all active historical data streams

Recording start/stop parameters:

-Alias <alias>	Alias name of the recording
-RecName <record>	Alias name of the data set
-Topic <topic>	Topic to be included
-DataFormat <Bin Json>	Data format of the live data stream
-Duration <seconds>	Recording duration
-Ringbuffer <None TimeBased DataBased>	Ring buffer mode (default value: <b>Default</b> )
-RinbufferPara <minutes/MB>	Parameters for the ring buffer (in seconds or megabytes)
-Mode <All Subset>	Mode of recording. Takes all symbols and a subset of the symbols
-Symbols / -Sym <Symbol1,Symbol2>	List of symbol subset as comma-separated list
-RecorderGuid <guid>	Guid of the Analytics Storage Provider Recorder window

#### Historical data stream start/stop parameters:

-SystemID <systemID guid>	System ID of the recorded data set
-Topic <topic>	Topic of the recorded data set
-Layout <layout guid>	Layout of the recorded data set
-RecordID <id>	ID of the data set to be streamed
-StartTime <time ns>	Start time of the data set to be streamed in nanoseconds
-EndTime <time ns>	End time of the data set to be streamed in nanoseconds
-MaxSamples <samples>	Maximum number of samples (default value: 5000)
-UsrSampleTime <ms>	Sampling rate. (Default value: -1; sampling rate of the recording)
-DataFormat <Bin Json>	Data format of the data stream
-ResultTopic <topic>	Result MQTT topic to which the data will be streamed
-Mode <All Subset>	Streaming mode. Streams all or a subset of the symbols
-Symbols / -Sym <Symbol1,Symbol2>	List of symbol subset as comma-separated list

#### Historical data stream update parameters:

-MaxSamples <samples>	Maximum number of samples (default value: 5000)
-UsrSampleTime <ms>	Sampling rate. (Default value: -1; sampling rate of the recording)
-MaxPackSize <samples>	Maximum message size in kilobytes
-SendDuration <ms>	Waiting time between sending messages in milliseconds
-ResultTopic <topic>	Result MQTT topic to which the data will be streamed

#### Command line samples:

##### Create configuration:

```
TwinCAT.Analytics.StorageProvider.Client
  -CreateASPConfig
    -MainTopic Beckhoff/ASPTest
    -Comment Analytics Storage Provider (Test)
    -EventLogTrace False
    -DebugLog False
    -StorageType ANALYTICSFILE
      -StorageConnString C:\TwinCAT\Functions\TF3520-Analytics-StorageProvider\Storage
      -MaxDuration 120
      -MaxWriteLen 2048
    -Broker 172.17.62.135
    -Port 1883
```

```
-User tcanalytics  
-Pwd 123
```

### Start recording with local Analytics Storage Provider:

```
TwinCAT.Analytics.StorageProvider.Client  
-localprovider  
-startrecord  
  -alias cmdTest  
  -recname cmdRec1  
  -topic TestSignals/TestStream  
  -dataformat Bin  
  -Duration 30  
  -mode Subset  
  -Symbols Variables.fCosine,Variables.fSine
```

### Start configuration file of a recording:

```
TwinCAT.Analytics.StorageProvider.Client  
-ConfigFile "C:  
\Users\User\AppData\Roaming\Beckhoff\TwinCAT Analytics Storage Provider\TcAnalyticsStorageProvider_R  
ecorder.xml"  
-ProviderGuid 76141a7f-e580-4281-99d8-1b8a75ca014d  
-startrecord  
-ConfigCmdAlias cmdTest
```

### Check recording status

```
TwinCAT.Analytics.StorageProvider.Client  
-Broker 172.17.62.135  
  -Port 1883  
  -User tcanalytics  
  -Pwd 123  
-ProviderGuid 76141a7f-e580-4281-99d8-1b8a75ca014d  
-IsRecordingActive  
  -alias cmdTest  
  -recorderGuid a8e171d2-712d-bd8e-da15-7eef28b71ad2
```

### Stop all recordings:

```
TwinCAT.Analytics.StorageProvider.Client  
-Broker 172.17.62.135  
  -Port 1883  
  -User tcanalytics  
  -Pwd 123  
-ProviderGuid 76141a7f-e580-4281-99d8-1b8a75ca014d  
-CancelAllRec
```

### Start historical data stream:

```
TwinCAT.Analytics.StorageProvider.Client  
-localprovider  
-GetHistorical  
  -systemID c29ac2d4-76ce-ff44-4d7f-355ffbcca6bf  
  -layout 9a8e171d-712d-bd8e-da15-7eef28b71ad2  
  -topic TestSignals/TestStream  
  -recordID 1  
  -startTime 132696863612730000  
  -endTime 13269686417720000  
  -maxSamples 5000  
  -usrSampleTime -1  
  -resultTopic _TestSignals/TestStream/123  
  -dataformat Bin  
  -mode Subset -symbols Variables.fSine
```

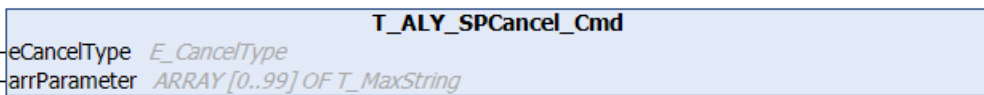
# 7 PLC API

## 7.1 Function blocks

### 7.1.1 Topic Architecture

#### 7.1.1.1 Commands

##### 7.1.1.1.1 T\_ALY\_SPCancel\_Cmd



#### Syntax

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 \[▶ 80\]](#)

T\_ALY\_SPCancel\_Cmd

#### 🔧 Inputs

Name	Type	Description
eCancelType	<a href="#">E_CancelType [▶ 88]</a>	
arrParameter	ARRAY [0..99] OF T_MaxString	

#### 🔗 Methods

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from <a href="#">T_ALY_JsonPayload [▶ 80]</a>	Initialize FB with Json object
Init_String	Inherited from <a href="#">T_ALY_JsonPayload [▶ 80]</a>	Initialize FB with Json string
GetJsonLength	Inherited from <a href="#">T_ALY_JsonPayload [▶ 80]</a>	Get Length of Json payload
GetJsonString	Inherited from <a href="#">T_ALY_JsonPayload [▶ 80]</a>	Get Json payload as String

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

### 7.1.1.1.2 T\_ALY\_SPGetHistorical\_Cmd

T_ALY_SPGetHistorical_Cmd	
—sTopic	T_MaxString
—sLayout	GUID
—eMode	E_SymbolMode
—eOutputFormat	E_RawDataFormat
—nMaxSampleCount	UDINT
—nUserSampleTime	DINT
—nRecordID	DINT
—nStartTimestamp	LINT
—nEndTimestamp	LINT
—sResultTopic	T_MaxString
—arrSymbols	ARRAY [0..255] OF T_ALY_Symbol

#### 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 \[▶ 80\]](#)

T\_ALY\_SPGetHistorical\_Cmd

**Inputs**

Name	Type	Description
sTopic	T_MaxString	Topic name of the recorded Live Stream
sLayout	GUID	Layout GUID of the recording
eMode	E_SymbolMode <a href="#"> &gt; 90</a>	Get all symbols or only a subset
eOutputFormat	E_RawDataFormat <a href="#"> &gt; 88</a>	Format of the returned data (actually only "Bin" supported)
nMaxSampleCount	UDINT	Max count of samples in one payload packet
nUserSampleTime	DINT	Sampletime in milliseconds of the returned stream. (-1 use the recorded sampletime)
nRecordID	DINT	Number of the record
nStartTimestamp	LINT	StartTime
nEndTimestamp	LINT	EndTime
sResultTopic	T_MaxString	Topicname of the result stream.
arrSymbol	ARRAY [0..255] OF T_ALY_Symbol <a href="#"> &gt; 78</a>	If SymbolMode is Subset, only the list of this symbols will be returned

**Methods**

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from T_ALY_JsonPayload <a href="#"> &gt; 80</a>	Initialize FB with Json object
Init_String	Inherited from T_ALY_JsonPayload <a href="#"> &gt; 80</a>	Initialize FB with Json string
GetJsonLength	Inherited from T_ALY_JsonPayload <a href="#"> &gt; 80</a>	Get Length of Json payload
GetJsonString	Inherited from T_ALY_JsonPayload <a href="#"> &gt; 80</a>	Get Json payload as String

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

**7.1.1.1.3 T\_ALY\_SPReadStreamRecords\_Cmd**

```

T_ALY_SPReadStreamRecords_Cmd
— sStreamTopic STRING(255)
— sStreamSystemID GUID
— sStreamLayout GUID
— nRecordStartIndex DINT
— nMaxRecordCount DINT
— sResultTopic T_MaxString
    
```

**Syntax**

**Definition:**

```
FUNCTION_BLOCK T_ALY_SPReadStreamRecords_Cmd EXTENDS T_ALY_JsonPayload
VAR_INPUT
    sStreamTopic : STRING(255);
    sStreamSystemID : GUID;
    sStreamLayout : GUID;
    nRecordStartIndex : DINT;
    nMaxRecordCount : DINT;
    sResultTopic : T_MaxString;
END_VAR
```

**Inheritance hierarchy**

[T\\_ALY\\_JsonPayload \[▶ 80\]](#)

T\_ALY\_SPReadStreamRecords\_Cmd

 **Inputs**

Name	Type	Description
sStreamTopic	STRING(255)	Topic name of the recorded live stream.
sStreamSystemID	GUID	SystemID of the target system from where the live stream was sent
sStreamLayout	GUID	Layout GUID of the recording
nRecordStartIndex	DINT	Start index of the first record to be read.
nMaxRecordCount	DINT	Total number of records to be read.
sResultTopic	T_MaxString	Topic name of the result stream

 **Methods**

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from <a href="#">T_ALY_JsonPayload [▶ 80]</a>	Initialize FB with Json object
Init_String	Inherited from <a href="#">T_ALY_JsonPayload [▶ 80]</a>	Initialize FB with Json string
GetJsonLength	Inherited from <a href="#">T_ALY_JsonPayload [▶ 80]</a>	Get Length of Json payload
GetJsonString	Inherited from <a href="#">T_ALY_JsonPayload [▶ 80]</a>	Get Json payload as String

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

### 7.1.1.1.4 T\_ALY\_SPRecordData\_Cmd

	T_ALY_SPRecordData_Cmd
—	sAlias <i>T_MaxString</i>
—	sRecordName <i>T_MaxString</i>
—	eRecording <i>E_RecordMode</i>
—	sRecorder <i>GUID</i>
—	sRecorderAlias <i>T_MaxString</i>
—	sTopic <i>T_MaxString</i>
—	eDataFormat <i>E_RawDataFormat</i>
—	nDuration <i>DINT</i>
—	eRingBufferMode <i>E_RingBufferMode</i>
—	nRingBufferParameter <i>DINT</i>
—	eMode <i>E_SymbolMode</i>
—	sSymbolLayout <i>GUID</i>
—	arrSymbols <i>ARRAY [0..255] OF T_ALY_Symbol</i>

#### 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](#) [► 80]

T\_ALY\_SPRecordData\_Cmd



 **Inputs**

Name	Type	Description
sAlias	T_MaxString	Alias name for the Recording
sRecordName	T_MaxString	Name for this record
eRecording	E_RecordMode [ <a href="#">▶ 88</a> ]	Start or Stop the recording
sRecorder	GUID	Individual GUID of the recorder
sRecorderAlias	T_MaxString	Alias name for the recorder
sTopic	T_MaxString	Topic name of the live stream
eDataFormat	E_RawDataFormat [ <a href="#">▶ 88</a> ]	Store data format. (actually only Binary format is supported)
nDuration	DINT	Duration in minutes of the recording. (-1 unlimited)
eRingBufferMode	E_RingBufferMode [ <a href="#">▶ 89</a> ]	Ringbuffer modus
nRingBufferParameter	DINT	TimeBased => Parameter in minutes DataBased => Parameter in Megabytes
eMode	E_SymbolMode [ <a href="#">▶ 90</a> ]	Record all symbols or only a subset
sSymbolLayout	GUID	
arrSymbols	ARRAY [0..255] OF T_ALY_Symbol [ <a href="#">▶ 78</a> ]	If SymbolMode is Subset, only the list of this symbols will be recorded

 **Methods**

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from T_ALY_JsonPayload [ <a href="#">▶ 80</a> ]	Initialize FB with Json object
Init_String	Inherited from T_ALY_JsonPayload [ <a href="#">▶ 80</a> ]	Initialize FB with Json string
GetJsonLength	Inherited from T_ALY_JsonPayload [ <a href="#">▶ 80</a> ]	Get Length of Json payload
GetJsonString	Inherited from T_ALY_JsonPayload [ <a href="#">▶ 80</a> ]	Get Json payload as String

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

**7.1.1.1.5 T\_ALY\_SPREloadHistoricalStreams\_Cmd**

**T\_ALY\_SPREloadHistoricalStreams\_Cmd**

—eReloadType *E\_ReloadType*  
 —arrParameter *ARRAY [0..9] OF ARRAY [0..1] OF T\_MaxString*

**Syntax**

Definition:

```
FUNCTION_BLOCK T_ALY_SPReloadHistoricalStreams_Cmd EXTENDS T_ALY_JsonPayload
VAR_INPUT
    eReloadType : E_ReloadType;
    arrParameter : ARRAY [0..9] OF ARRAY [0..1] OF T_MaxString;
END_VAR
```

**Inheritance hierarchy**

T\_ALY\_JsonPayload [▶ 80]

T\_ALY\_SPReloadHistoricalStreams\_Cmd

 **Inputs**

Name	Type	Description
eReloadType	E_ReloadType [▶ 89]	Update mode selection
arrParameter	ARRAY [0..9] OF ARRAY [0..1] OF T_MaxString	Additional parameters

 **Methods**

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from <u>T_ALY_JsonPayload</u> [▶ 80]	Initialize FB with Json object
Init_String	Inherited from <u>T_ALY_JsonPayload</u> [▶ 80]	Initialize FB with Json string
GetJsonLength	Inherited from <u>T_ALY_JsonPayload</u> [▶ 80]	Get Length of Json payload
GetJsonString	Inherited from <u>T_ALY_JsonPayload</u> [▶ 80]	Get Json payload as String

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

**7.1.1.1.6 T\_ALY\_SPSetGetHistoricalDataState\_Cmd**

**T\_ALY\_SPSetGetHistoricalDataState\_Cmd**

- sResultTopic *T\_MaxString*
- eState *E\_SetGetHistoricalDataState*
- nSendDuration\_ms *DINT*
- nRestartTimestamp *LINT*
- nMaxSampleCount *UDINT*
- nMaxPackageSize\_KB *DINT*
- nUserSampleTime *LINT*

**Syntax**

Definition:

```
FUNCTION_BLOCK T_ALY_SPSetGetHistoricalDataState_Cmd EXTENDS T_ALY_JsonPayload
VAR_INPUT
    sResultTopic : T_MaxString;
    eState : E_SetGetHistoricalDataState;
    nSendDuration_ms : DINT;
    nRestartTimestamp : LINT;
    nMaxSampleCount : UDINT;
    nMaxPackageSize_KB: DINT;
    nUserSampleTime : LINT;
END_VAR
```

**Inheritance hierarchy**

T\_ALY\_JsonPayload [[▶ 80](#)]

T\_ALY\_SPSetGetHistoricalDataState\_Cmd

 **Inputs**

Name	Type	Description
sResultTopic	T_MaxString	Topic name of the result stream (used like a handle).
eState	<a href="#">E_SetGetHistoricalDataState ▶ 89</a>	Historical stream state
nSendDuration_ms	DINT	Waiting time between sending the individual packages
nRestartTimestamp	LINT	Timestamp at which the result stream is continued.
nMaxSampleCount	UDINT	Maximum number of entries in a package
nMaxPackageSize_KB	DINT	Maximum size of a package
nUserSampleTime	LINT	Sample time in milliseconds of the returned stream (-1 uses the recorded sample time).

 **Methods**

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from <a href="#">T_ALY_JsonPayload ▶ 80</a>	Initialize FB with Json object
Init_String	Inherited from <a href="#">T_ALY_JsonPayload ▶ 80</a>	Initialize FB with Json string
GetJsonLength	Inherited from <a href="#">T_ALY_JsonPayload ▶ 80</a>	Get Length of Json payload
GetJsonString	Inherited from <a href="#">T_ALY_JsonPayload ▶ 80</a>	Get Json payload as String

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 Descriptions

### 7.1.1.2.1 T\_ALY\_HistoricalStream\_Desc



#### Syntax

##### Definition:

```

FUNCTION_BLOCK T_ALY_HistoricalStream_Desc EXTENDS T_ALY_JsonPayload
VAR_OUTPUT
    sSource : STRING(255);
    sStreamTopic : STRING(255);
    sStreamAlias : STRING(255);
    sStreamSystemID : GUID;
    sLayout : GUID;
    nCycleTime: UDINT;
    nDataSize : UDINT;
    arrRecords : ARRAY [0..cMaxRecordCount] OF T_RecordTimestamps;
END_VAR
    
```

#### Inheritance hierarchy

T\_ALY\_JsonPayload [[▶ 80](#)]

    T\_ALY\_HistoricalStream\_Desc

#### Outputs

Name	Type	Description
sSource	STRING(255)	Data source name
sStreamTopic	STRING(255)	Topic name of the recorded stream
sStreamAlias	STRING(255)	Alias name of the stream
sStreamSystemID	GUID	SystemID GUID of the stream
sLayout	GUID	Layout GUID of the recording
nCycleTime	UDINT	Cycle time of the recording
nDataSize	UDINT	Data size of an entry of the recording
arrRecords	ARRAY [0..cMaxRecordCount] OF T_RecordTimestamps [ <a href="#">▶ 79</a> ]	Timestamp of the various recordings

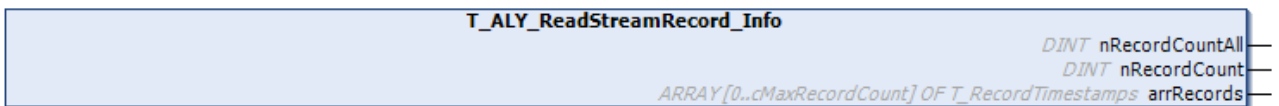
 **Methods**

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">▶ 80</a> ]	Initialize FB with Json object
Init_String	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">▶ 80</a> ]	Initialize FB with Json string
GetJsonLength	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">▶ 80</a> ]	Get Length of Json payload
GetJsonString	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">▶ 80</a> ]	Get Json payload as String

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

**7.1.1.3 Info**

**7.1.1.3.1 T\_ALY\_ReadStreamRecord\_Info**



**Syntax**

Definition:

```

FUNCTION_BLOCK T_ALY_HistoricalStream_Desc EXTENDS T_ALY_JsonPayload
VAR_OUTPUT
    nRecordCountAll: UDINT;
    nRecordCount : UDINT;
    arrRecords : ARRAY [0..cMaxRecordCount] OF T_RecordTimestamps;
END_VAR
    
```

**Inheritance hierarchy**

[T\\_ALY\\_JsonPayload](#) [[▶ 80](#)]

T\_ALY\_ReadStreamRecord\_Info

 **Outputs**

Name	Type	Description
nRecordCountAll	UDINT	Number of all existing records
nRecordCount	UDINT	Number of records read out
arrRecords	ARRAY [0..cMaxRecordCount] OF <a href="#">T_RecordTimestamps</a> [ <a href="#">▶ 79</a> ]	Timestamp of the records read out

**Methods**

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">▶ 80</a> ]	Initialize FB with Json object
Init_String	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">▶ 80</a> ]	Initialize FB with Json string
GetJsonLength	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">▶ 80</a> ]	Get Length of Json payload
GetJsonString	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">▶ 80</a> ]	Get Json payload as String

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

### 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](#) [[▶ 80](#)]

T\_ALY\_Symbol

**Inputs**

Name	Type	Description
sName	T_MaxString	Name of the symbol
sBaseType	T_MaxString	DataType of the symbol
nBitOffset	UDINT	BitOffset of the symbol
nBitSize	UDINT	BitSize of the symbol

Methods

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">80</a> ]	Initialize FB with Json object
Init_String	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">80</a> ]	Initialize FB with Json string
GetJsonLength	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">80</a> ]	Get Length of Json payload
GetJsonString	Inherited from <a href="#">T_ALY_JsonPayload</a> [ <a href="#">80</a> ]	Get Json payload as String

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

### 7.1.1.4.2 T\_RecordTimestamps



#### Syntax

Definition:

```

FUNCTION_BLOCK T_RecordTimestamps EXTENDS T_ALY_JsonPayload
VAR_OUTPUT
  nRecordID : DINT;
  sAlias : STRING(255);
  nStartTimestamp : LINT;
  nEndTimestamp : LINT;
END_VAR
    
```

#### Inheritance hierarchy

[T\\_ALY\\_JsonPayload](#) [[80](#)]

T\_RecordTimestamps

#### Outputs

Name	Type	Description
nRecordID	DINT	Recording number
sAlias	STRING(255)	Alias name of the recording
nStartTimestamp	LINT	Start timestamp of the recording
nEndTimestamp	LINT	End timestamp of the recording

**Methods**

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from <a href="#">T_ALY_JsonPayload</a> [▶ 80]	Initialize FB with Json object
Init_String	Inherited from <a href="#">T_ALY_JsonPayload</a> [▶ 80]	Initialize FB with Json string
GetJsonLength	Inherited from <a href="#">T_ALY_JsonPayload</a> [▶ 80]	Get Length of Json payload
GetJsonString	Inherited from <a href="#">T_ALY_JsonPayload</a> [▶ 80]	Get Json payload as String

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

### 7.1.1.5 Base Types

#### 7.1.1.5.1 T\_ALY\_JsonPayload



#### Syntax

Definition:

```
FUNCTION_BLOCK INTERNAL T_ALY_JsonPayload
```

**Methods**

Name	Definition location	Description
Reset		Reset all values inside of the payload FB
Init_JsonValue	Inherited from <a href="#">T_ALY_JsonPayload</a> [▶ 80]	Initialize FB with Json object
Init_String	Inherited from <a href="#">T_ALY_JsonPayload</a> [▶ 80]	Initialize FB with Json string
GetJsonLength	Inherited from <a href="#">T_ALY_JsonPayload</a> [▶ 80]	Get Length of Json payload
GetJsonString	Inherited from <a href="#">T_ALY_JsonPayload</a> [▶ 80]	Get Json payload as String

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



## 7.1.2 FB\_ALY\_StorageProvider



The FB\_ALY\_StorageProvider is a client FB for communication with a Storage Provider instance. The FB provides methods to trigger historical data or start/stop recordings.

### Syntax

Definition:

```
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

Name	Type	Description
stConfig	ST_ALY_SP_Config [ <a href="#">▶ 87</a> ]	Structure for the configuration of the FB.

### Outputs

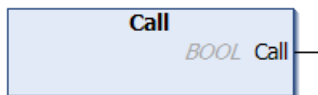
Name	Type	Description
bBusy	BOOL	TRUE as soon as a method of the function block is active.
bError	BOOL	Becomes TRUE when an error situation occurs.
ipResultMessage	I_TcMessage	Message interface of the TwinCAT 3 EventLogger, which provides further information about the return value.
eConnectionState	ETcIotMqttClientState	Specifies the state of the connection between client and broker as an enumeration ETcIotMqttClientState.

 **Methods**

Name	Definition location	Description
<a href="#">Call</a> [ <a href="#">▶ 82</a> ]	Local	Method for background communication with the TwinCAT driver. The method must be called cyclically.
<a href="#">Cancel</a> [ <a href="#">▶ 82</a> ]	Local	Method for aborting activities of the TwinCAT Analytics Storage Provider.
<a href="#">GetHistoricalData</a> [ <a href="#">▶ 83</a> ]	Local	Method for requesting historical data.
<a href="#">ReadHistoricalStreams</a> [ <a href="#">▶ 83</a> ]	Local	Method for reading all historical streams.
<a href="#">ReadStreamRecords</a> [ <a href="#">▶ 84</a> ]	Local	Method for reading all records of a historical stream
<a href="#">ResetCommunication</a> [ <a href="#">▶ 84</a> ]	Local	Method to reset the MQTT connection to the broker.
<a href="#">SendCommand</a> [ <a href="#">▶ 85</a> ]	Local	Generic method for sending various commands.
<a href="#">SetHistoricalDataState</a> [ <a href="#">▶ 85</a> ]	Local	Method for setting various parameters of a historical stream
<a href="#">StartRecord</a> [ <a href="#">▶ 86</a> ]	Local	Starts recording a live MQTT binary stream.
<a href="#">StopRecord</a> [ <a href="#">▶ 86</a> ]	Local	Stops the selected recording.

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

**7.1.2.1 Call**



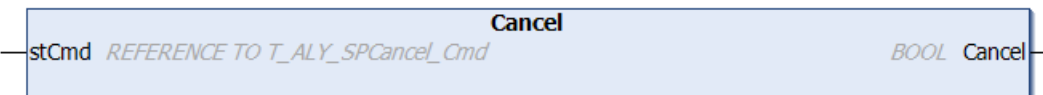
**Syntax**

METHOD Call : BOOL

 **Return value**

Name	Type	Description
Call	BOOL	

**7.1.2.2 Cancel**



**Syntax**

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

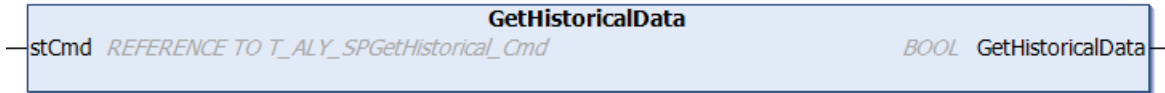
 **Inputs**

Name	Type	Description
stCmd	REFERENCE TO <a href="#">T_ALY_SPCancel_Cmd</a> [ <a href="#">▶ 68</a> ]	Json command to cancel operations of the TwinCAT Analytics Storage Provider

 Return value

Name	Type	Description
Cancel	BOOL	Is TRUE if done

### 7.1.2.3 GetHistoricalData



**Syntax**

```
METHOD GetHistoricalData : BOOL
VAR_INPUT
    stCmd : REFERENCE TO T_ALY_SPHistorical_Cmd;
END_VAR
```

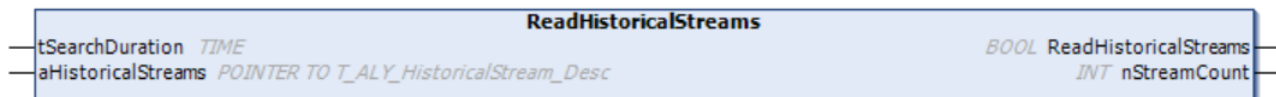
 Inputs

Name	Type	Description
stCmd	REFERENCE TO T_ALY_SPGetHistorical_Cmd <a href="#">▶ 69</a>	Json command to get historical data from TwinCAT Analytics Storage Provider

 Return value

Name	Type	Description
GetHistoricalData	BOOL	Is TRUE if done

### 7.1.2.4 ReadHistoricalStreams



**Syntax**

```
METHOD GetHistoricalData : BOOL
VAR_INPUT
    tSearchDuration : TIME := TIME#5s0ms
    aHistoricalStreams : POINTER TO T_ALY_HistoricalStream_Desc;
    nStreamCount : INT
END_VAR
```

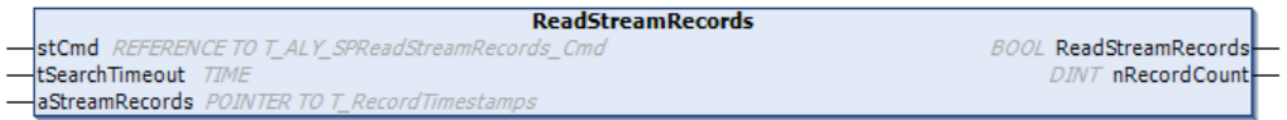
 Inputs

Name	Type	Description
tSearchDuration	TIME	Time period in which to wait for feedback.
aHistoricalStreams	POINTER TO T_ALY_HistoricalStream_Desc <a href="#">▶ 76</a>	Description of the different historical streams

 **Return value**

Name	Type	Description
ReadHistoricalStreams	BOOL	Is TRUE when completed
nStreamCount	INT	Number of streams read out

### 7.1.2.5 ReadStreamRecords



**Syntax**

```
METHOD ReadStreamRecords : BOOL
VAR_INPUT
    stCmd : REFERENCE TO T_ALY_SPReadStreamRecords_Cmd;
tSearchTimeout : TIME := TIME#5s0ms;
aStreamRecords : POINTER TO T_RecordTimestamps;
nRecordCount : DINT;
END_VAR
```

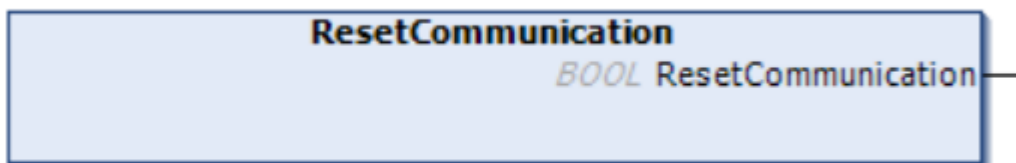
 **Inputs**

Name	Type	Description
stCmd	REFERENCE TO <u>T_ALY_SPReadStreamRecords_Cmd</u> [▶ 70]	Json command to retrieve recordings of a historical stream from TwinCAT Analytics Storage Provider
tSearchTimeout	TIME	Waiting time for the response
aStreamRecords	POINTER TO <u>T_RecordTimestamps</u> [▶ 79]	Recordings read out

 **Return value**

Name	Type	Description
ReadStreamRecords	BOOL	Is TRUE when completed
nRecordCount	DINT	Number of records read out

### 7.1.2.6 ResetCommunication



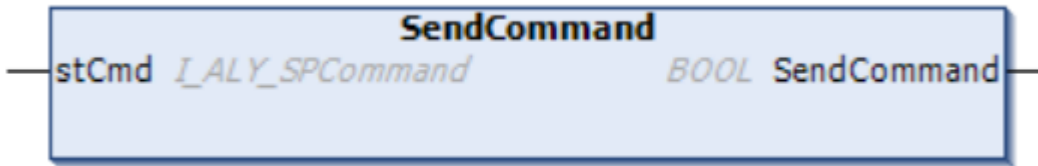
**Syntax**

```
METHOD ResetCommunication : BOOL
VAR_INPUT
END_VAR
```

 Return value

Name	Type	Description
ResetCommunication	BOOL	Is TRUE when completed

### 7.1.2.7 SendCommand



**Syntax**

```
METHOD SendCommand : BOOL
VAR_INPUT
    stCmd : I_ALY_SPCCommand;
END_VAR
```

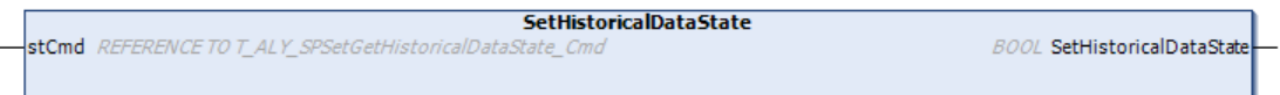
 Inputs

Name	Type	Description
stCmd	I_ALY_SPCCommand	Json command to interact with the TwinCAT Analytics Storage Provider

 Return value

Name	Type	Description
SendCommand	BOOL	Is TRUE when completed

### 7.1.2.8 SetHistoricalDataState



**Syntax**

```
METHOD SetHistoricalDataState : BOOL
VAR_INPUT
    stCmd : REFERENCE TO T_ALY_SPGetSetHistoricalDataState_Cmd;
END_VAR
```

 Inputs

Name	Type	Description
stCmd	REFERENCE TO <u>T_ALY_SPGetSetHistoricalDataState_Cmd</u> [ <a href="#">▶ 74</a> ]	Json command to set parameters of a started historical stream from the TwinCAT Analytics Storage Provider

 **Return value**

Name	Type	Description
SetHistoricalDataState	BOOL	Is TRUE when completed

### 7.1.2.9 StartRecord



**Syntax**

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

 **Inputs**

Name	Type	Description
stCmd	REFERENCE TO T_ALY_SPRecordData_Cmd [ <a href="#">▶ 72</a> ]	Json command to start recording a live stream

 **Return value**

Name	Type	Description
StartRecord	BOOL	Is TRUE if done

### 7.1.2.10 StopRecord



**Syntax**

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

 **Inputs**

Name	Type	Description
stCmd	REFERENCE TO T_ALY_SPRecordData_Cmd [ <a href="#">▶ 72</a> ]	Json command to stop recording of a live stream

 **Return value**

Name	Type	Description
StopRecord	BOOL	Is TRUE if done

## 7.2 Data types

### 7.2.1 ST\_ConnectionSettings

#### Syntax

Definition:

```

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

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

### 7.2.2 ST\_ALY\_SP\_Config

#### Syntax

Definition:

```

TYPE ST_Msg :
STRUCT
  sMainTopic      : T_MaxString;
  sProviderGuid   : GUID;
  stConnSettings : ST_ConnectionSettings
END_STRUCT
END_TYPE
    
```

**Parameter**

Name	Type	Description
sMainTopic	T_MaxString	The main topic where the TwinCAT Analytics Storage Provider is located on the message broker
sProviderGuid	GUID	The individual GUID of the TwinCAT Analytics Storage Provider Instance
stConnSettings	<a href="#">ST_ConnectionSettings</a> [► 87]	MQTT connection settings to connect with the message broker

**7.2.3 E\_CancelType****Syntax**

Definition:

```

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

```

**Parameter**

Name	Description
HistoricalData	Canceled the selected historical data stream
AllRecordData	Canceled all running recordings

**7.2.4 E\_RawDataFormat****Syntax**

Definition:

```

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

```

**Parameter**

Name	Description
Bin	Analytics Binary Stream Format
Json	TwinCAT Json Format (actually not supported)

**7.2.5 E\_RecordMode****Syntax**

Definition:

```

TYPE E_RecordMode :
(
    Start := 0,
    Stop
) INT;
END_TYPE

```



**Parameter**

Name	Description
Start	Starts the recording of the configured record
Stop	Stops the recording

## 7.2.6 E\_ReloadType

**Syntax**

Definition:

```

TYPE E_ReloadType :
(
    All := 0,
    Specific
) INT;
END_TYPE
    
```

**Parameters**

Name	Description
All	All records are read in again.
Specific	Only one specific record will be reread.

## 7.2.7 E\_RingBufferMode

**Syntax**

Definition:

```

TYPE E_RingBufferMode:
(
    None := 0,
    TimeBased,
    DataBased
) INT;
END_TYPE
    
```

**Parameter**

Name	Description
None	Recording without ringbuffer mode
TimeBased	Ringbuffer based on a given time periode
DataBased	Ringbuffer based on a given max data size

## 7.2.8 E\_SetGetHistoricalDataState

**Syntax**

Definition:

```

TYPE E_SetGetHistoricalDataState :
(
    Pause,
    Continue_,
    Restart,
    Stop,
    Update
) INT;
END_TYPE
    
```

**Parameters**

Name	Description
Break	Playback of the recording is paused.
Continue_	Playback of the recording continues.
Restart	Playback of the recording is restarted.
Stop	Playback of the recording is stopped.
Update	Parameters for playing the recording are updated.

**7.2.9 E\_SymbolMode****Syntax**

Definition:

```

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

```

**Parameter**

Name	Description
All	All symbols of the stream will be used
Subset	Only a subset of symbols will be used

## 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:

1. Analytics Logger: stream of variables from a Global Variable List to a MQTT Message Broker.
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

**About TwinCAT System**

TwinCAT System Service v3.1.0.2434 OK

**TwinCAT v3.1.4024.22**

Copyright BECKHOFF Automation © 1996-2021

AMS Net Id:

Logon User:  HW Platform:

User Group:  **System Id:**

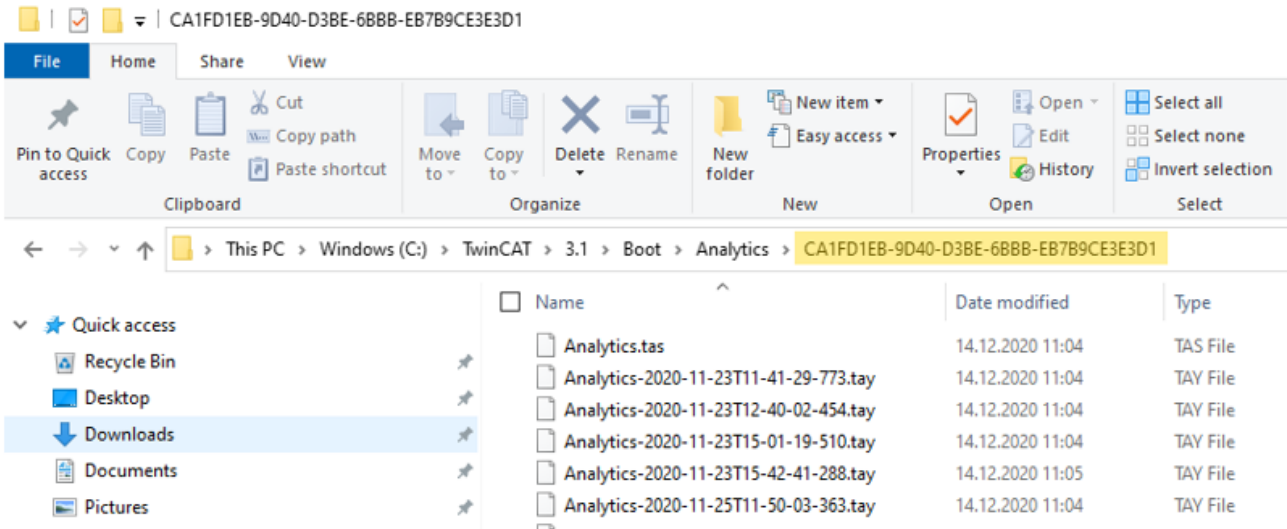
Self Signed Certificate - Fingerprint:

Licenses:

Order No	License	Instances	Status
	TC3 Analytics - Math	cpu license	valid
	TE3500 Analytics Workbench Ju...	cpu license	valid
	TC3 Analytics - IoT	cpu license	valid
	TC3 MMS	cpu license	valid
	TC3 Analytics - XY Path Analysis	cpu license	valid
	TC3 Condition Monitoring Base...	cpu license	valid

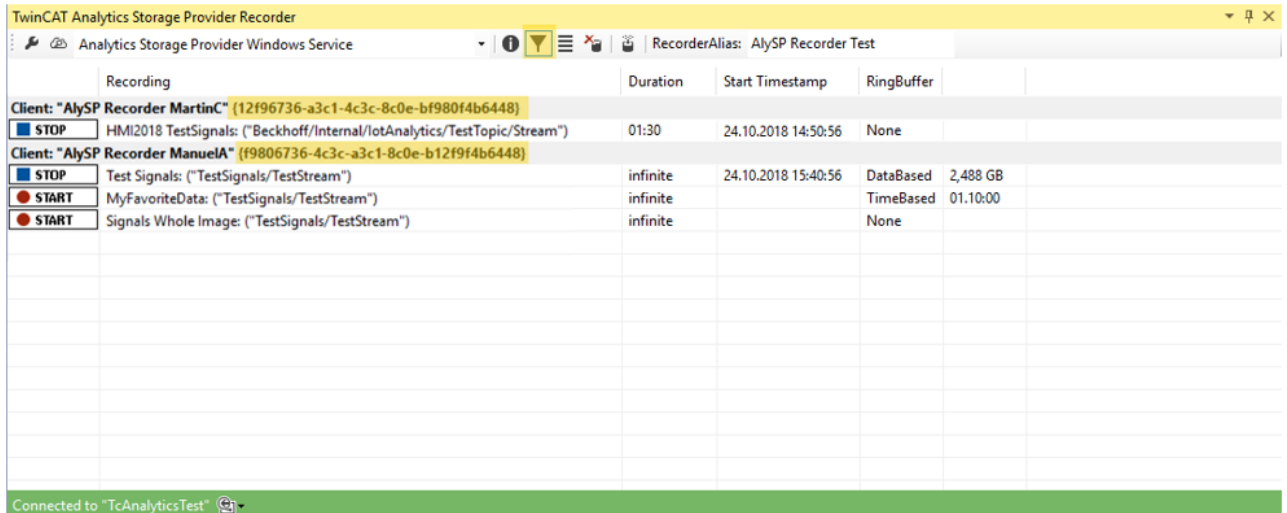
#### Symbol Info GUID

Identification ID for the symbols from the TwinCAT Analytics binary format.



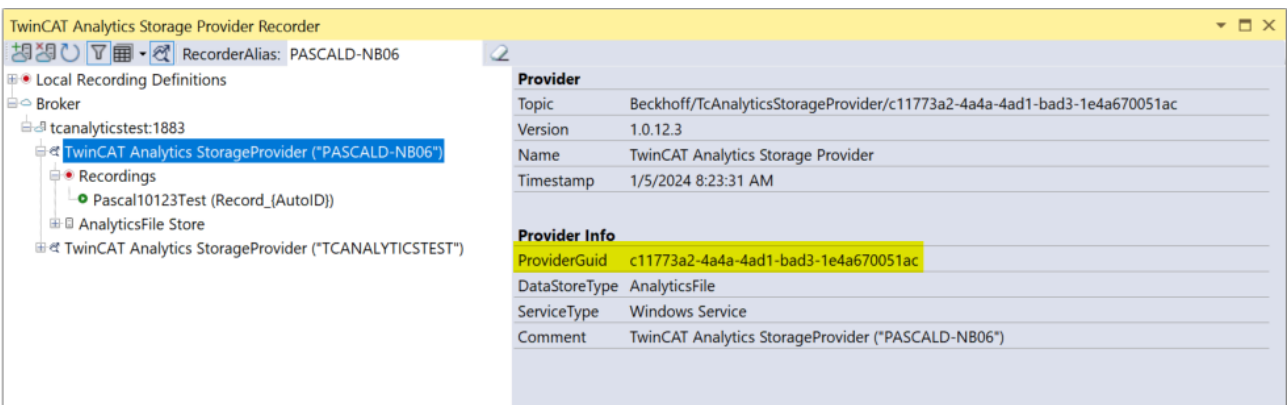
**Storage Provider Recorder GUID**

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



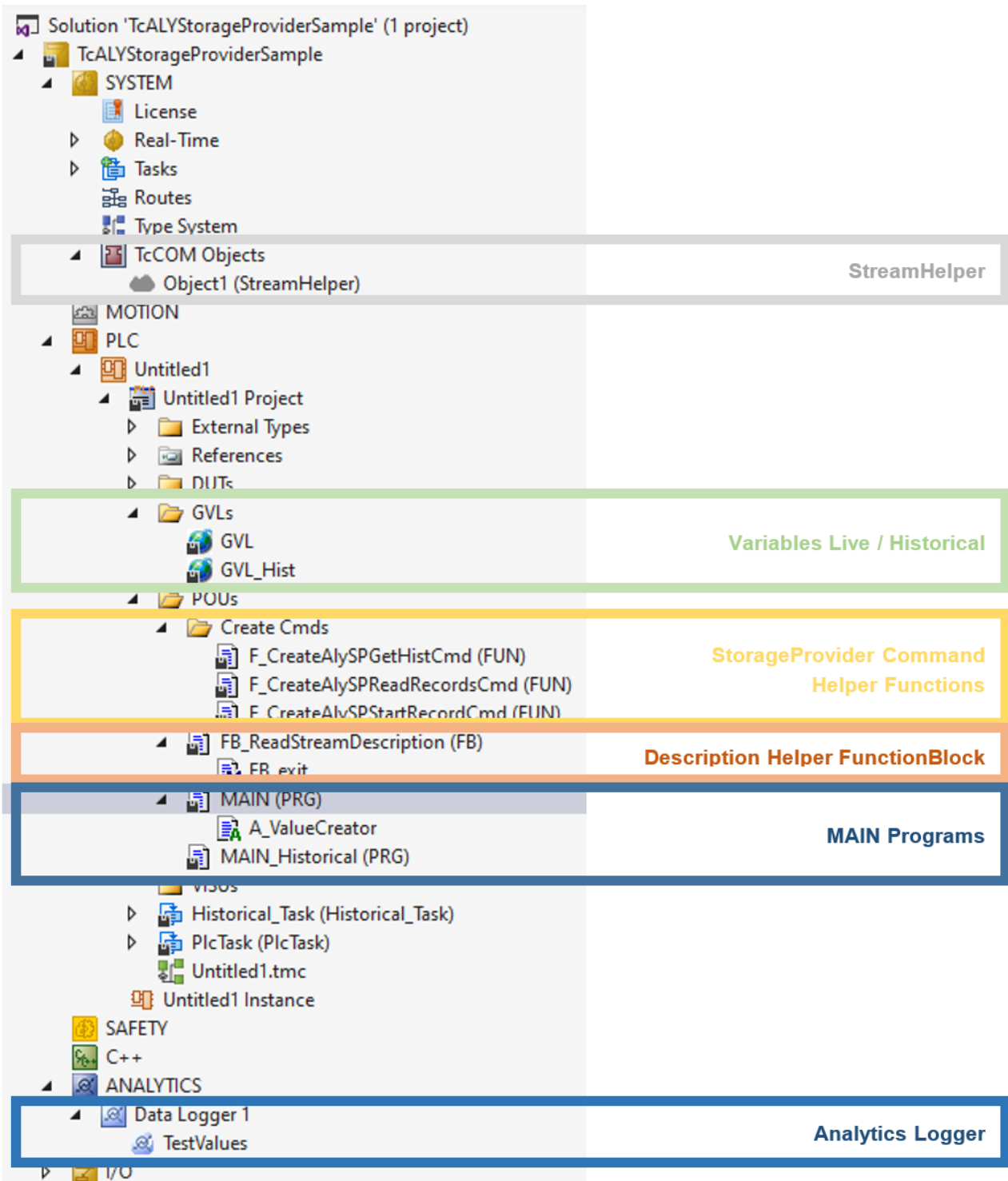
**Storage Provider Service GUID**

GUID of the individually used Storage Provider Service. If you are connected to a Storage Provider in the Recorder window, you can find the Provider GUID in the properties.



**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.

**IsRecordingActive**

This shows you how to parse the description sent by the Analytics Storage Provider using the JsonXML library, for example, and obtain information as to whether a specified Record Alias name is currently being recorded.

**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:

Object	Context	Parameter (Init)	Parameter (Online)	Interfaces		
		<b>Name</b>	<b>Value</b>	<b>CS</b>	<b>Type</b>	<b>PTCID</b>
-		MQTT				
		Host Name	172.17.62.145	<input type="checkbox"/>	STRING(80)	0x020201...
		TcpPort	1883	<input type="checkbox"/>	UINT	0x020201...
		Topic Prefix	_AlySPTest/ResultValues	<input type="checkbox"/>	STRING(255)	0x020300...
		Client ID		<input type="checkbox"/>	STRING(80)	0x020201...
		Username		<input type="checkbox"/>	STRING(80)	0x020201...
		Password		<input type="checkbox"/>	STRING(80)	0x020201...
		Max Handles	100	<input type="checkbox"/>	UDINT	0x020300...
		Fifo Exponent	8	<input type="checkbox"/>	UDINT	0x020300...
		Sender System Id		<input type="checkbox"/>	STRING(80)	0x020300...
-		Tls				
		CaFile		<input type="checkbox"/>	STRING(255)	0x020201...
		CertFile		<input type="checkbox"/>	STRING(255)	0x020201...
		KeyFile		<input type="checkbox"/>	STRING(255)	0x020201...
		KeyPw		<input type="checkbox"/>	STRING(255)	0x020201...
		CrlFile		<input type="checkbox"/>	STRING(255)	0x020201...
		PskId		<input type="checkbox"/>	STRING(255)	0x020201...
		PskPw		<input type="checkbox"/>	STRING(255)	0x020201...
		Insecure	FALSE	<input type="checkbox"/>	BOOL	0x020201...
		Version		<input type="checkbox"/>	STRING(255)	0x020201...

Analytics Logger:

Context	Parameter (Int)	Data Area	TLS	Time Source		
	Name	Value	CS	Type	PTCID	
	Data Format	ANALYTICS_FORMAT_MQTT_BINARY	<input type="checkbox"/>	ANALYTICS_FORMAT	0x02020114	
	Data Compression	ANALYTICS_COMPRESSION_RL	<input type="checkbox"/>	ANALYTICS_COMPRESS...	0x02030027	
	Max. Compression Compare Width	ANALYTICS_COMP_WIDTH_8	<input type="checkbox"/>	ANALYTICS_COMPRESS...	0x02030030	
	MQTT Host Name	172.17.62.145	<input type="checkbox"/>	STRING(80)	0x02020113	
	MQTT Tcp Port	1883	<input type="checkbox"/>	UINT	0x02020103	
	MQTT Main Topic	AlySPTest	<input type="checkbox"/>	STRING(255)	0x02030008	
	MQTT Client ID		<input type="checkbox"/>	STRING(80)	0x02020101	
	MQTT User Name		<input type="checkbox"/>	STRING(255)	0x02020106	
	MQTT Password		<input type="checkbox"/>	STRING(80)	0x02020107	

MAIN program:

```

MAIN
VAR
  nState: INT;
  fbALY_StorageProvider: FB_ALY_StorageProvider := (stConfig := (sMainTopic := 'Beckhoff', sProviderGuid := STRING_TO_GUID('1f6e80ac-5250-4a2e-a105-c0c2d9a981fd'),
  stConnSettings := (sHostName := '172.17.62.145', sUserID := '', sPassword := '', sClientId := 'PLCClient')));
  fbReadStreamDescription: FB_ReadStreamDescription := (stConnSettings := (sHostName := '172.17.62.145', sUserID := '', sPassword := '', sClientId := 'PLCClient2'));
  stStartCmd: T_ALY_SPRecordData_Cmd;
  stStopCmd: T_ALY_SPRecordData_Cmd;
  stReadRecordsCmd: T_ALY_SPReadStreamRecords_Cmd;
  stGetHistDataCmd: T_ALY_SPGetHistorical_Cmd;
  aRecordInfos: ARRAY [1..10] OF T_RecordTimestamps;
  nRecs: DINT;
END_VAR

```

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.

The Storage Provider Recorder GUID selected in the document above can optionally be set in the PLC in the F\_CreateAlySPStartRecordCmd function. Theoretically, it can be any GUID, it is only used to identify the recorder - in this case the PLC. However, if you set the Recorder GUID from your own engineering, the record is displayed directly in the Analytics Storage Provider Recorder. If a different GUID is used, the filter function in the Recorder window may first have to be activated so that the record from the PLC is displayed.

Download: [https://infosys.beckhoff.com/content/1033/tf3500\\_tc3\\_analytics\\_logger/Resources/11270100747.zip](https://infosys.beckhoff.com/content/1033/tf3500_tc3_analytics_logger/Resources/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](mailto:support@beckhoff.com).

1. [How can I manage the table schema of MS SQL with ASP? \[► 96\]](#)
2. [Can I control the Storage Provider in a programmable way? \[► 96\]](#)
3. [Is it also possible to save results from the Analytics Runtime? \[► 96\]](#)
4. [Are open source software components used in TwinCAT Measurement products? \[► 96\]](#)
5. [What factors influence the data throughput of the storage provider? \[► 96\]](#)

#### 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 \[► 96\]](#).

#### What factors influence the data throughput of the storage provider?

The data throughput depends on many influencing variables. Primarily of system and network resources. An overview:

- System properties (CPU, RAM)
- Writing speed and quality of the storage medium (SSD)
- Network properties
- Complexity of symbolism (data type, structures, arrays, etc.)
- Mode of historization (total symbolism allows higher throughput, a subset may be more costly depending on its size)
- Compression level of the stream (the stronger the compression, the higher the system load)
- The size of a sample
- Total size of a data packet (number of samples per packet)
- The number of parallel recordings that the storage provider manages

### 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*





More Information:  
[www.beckhoff.com/tf3520/](http://www.beckhoff.com/tf3520/)

Beckhoff Automation GmbH & Co. KG  
Hülshorstweg 20  
33415 Verl  
Germany  
Phone: +49 5246 9630  
[info@beckhoff.com](mailto:info@beckhoff.com)  
[www.beckhoff.com](http://www.beckhoff.com)

