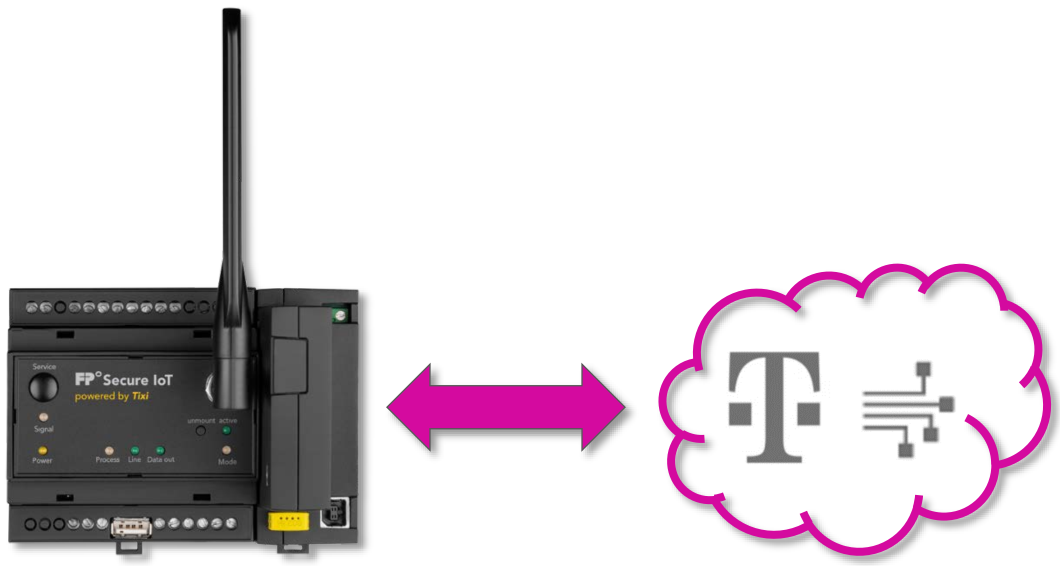
Telekom CoT Client for FP Gateways
User Manual



Version: 1.0.2

Publication date: 23/02/2021

Table of contents

1 Introduction

As of firmware version 5.2.6.36, FP gateways can connect to the “Cloud der Dinge” (cloud of things, CoT) from Deutsche Telekom. For more information regarding the “Cloud der Dinge” (cloud of things), see here:
https://iot.telekom.com/iot-de/plattformen/cloud-der-dinge

You can obtain free test access to the “Cloud der Dinge” (cloud of things) via the following link:
https://support.ram.m2m.telekom.com/apps/demotenant/index.html#/

1.1 TLS encryption

The connection to the broker is always established encrypted (port 8883). Unencrypted connections are not possible in the Deutsche Telekom network for security reasons.

1.2 Sending interval

As soon as an MQTT connection has been established, the client sends process data to the Telekom broker regularly. The sending interval can be adjusted using the parameter (RTSendTimeX).

2 TiXML configuration for the CoT client

The CoT client is configured using a TiXML database.

```
[<SetConfig _="ISP" ver="y">

  <CloudConn>

    <!-- connection with tls/ssl -->
    <!-- replace nb-iot.ram.m2m.telekom.com with the URL given
         from Deutsche Telekom for your CoT tenant -->
    <CloudBaseUrl _="ssl://nb-iot.ram.m2m.telekom.com:8883"/>

    <!-- if set to 1 the MQTT connection will be started at device
         startup; otherwise MQTT connection will not be established -->
    <CloudConnStart _="1"/>

    <!-- fixed setting (mandatory) -->
    <CloudConnType _="CoT_MQTTConnector_1"/>

    <!-- RTSendTimeX and UpdateRate1 controls the send interval
          X = 1 .. 5
          RTSendTimeX ranges:
           RTSendTimeX = -1: do not send data using a time
                             or activate hysteresis (RTSendTime1 only)
           RTSendTimeX = 1 .. 2^30: send interval in seconds
           RTSendTimeX = empty: send interval = 3 seconds (default)

          UpdateRate1 ranges (only used when RTSendTime1 = -1)
           UpdateRate1 = 1 .. 3600 = update interval of data
                                     points in seconds

          Hysteresis is defined within the RealTime1 section (below).
          In hysteresis mode the variables will only be sent to the
          broker if a pre-defined treshhold is reached
          After a connection to the broker has been established all
          variables of a hysteris RealTime1-section will be sent once.

          hysteresis ranges: floating point number, max. two digits
          after the decimal point. Examples: 0.1 / 2 / 2.68
     -->

     <!-- specifies the first send timer, operating mode = hysteresis -->
     <RTSendTime1 _="-1" />

     <!-- update rate for variables in hysteresis mode in seconds -->
     <UpdateRate1 _="20" />

     <!-- list of variables for first send timer to be sent to broker
          The gateway reads out the current values from the Process
          tree using the UpdateRateX. If a data point has changed and
          the changes are bigger than the specified hysteresis the data
          point will be sent to the broker.
     -->
     <Realtime1>
       <Datapoint1 _="/Process/Bus1/Device1/Datapoint1" hysteresis="2"/>
       <Datapoint2 _="/Process/Bus1/Device1/Datapoint2" hysteresis="1"/>
       <Datapoint3 _="/Process/Bus1/Device2/Datapoint3" hysteresis="5"/>
       <Datapoint4 _="/Process/PV/ProcVar1" hysteresis="0.4"/>
       <Datapoint5 _="/Process/PV/ProcVar2" hysteresis="1.7"/>
     </Realtime1>

     <!-- specifies the second send timer, send interval = 10 seconds -->
     <RTSendTime2 _="10" />
     <Realtime2>
        <Obis_180 _="/Process/Meter/MT174/Obis_180"/>
     </Realtime2>

     <!-- specifies the third send timer, no send interval -->
     <!-- data can be sent event driven using CloudSendRealtimeData -->
     <RTSendTime3 _="-1" />
     <Realtime3>
        <Obis_181 _="/Process/Meter/MT174/Obis_181"/>
     </Realtime3>

     <!-- specifies the forth send timer, send interval = 60 seconds -->
     <RTSendTime4 _="60" />
     <Realtime4>
        <Obis_182 _="/Process/Meter/MT174/Obis_182"/>
     </Realtime4>

     <!-- specifies the fifth send timer, send interval = 100 seconds -->
     <RTSendTime5 _="100" />
     <Realtime5>
        <Temp_Board _="/Process/OneWire/Mainboard/TEMP_Board"/>
     </Realtime5>
     .
     .
     <!-- specifies the fifth send timer, send interval = 100 seconds -->
     <RTSendTime5 _="100" />
     <Realtime5>
        <Temp_Board _="/Process/OneWire/Mainboard/TEMP_Board"/>
     </Realtime5>

   </CloudConn>

</SetConfig>]
```

Please note the following information:

- The configuration options shown in bold must be configured (mandatory).
- A CloudBaseUrl must be configured.
- The path for the URL is:
  `ssl://nb-iot.server.m2m.telekom.com:8883`
  where **`nb-iot.server.m2m.telekom.com`** corresponds to the URL for your CoT server assigned by Telekom.
  If in doubt, ask Deutsche Telekom for the complete URL.
- CloudConnStart and CloudConnType must be configured as described above.
- The data in the branches Realtime1 - RealTime5 can be configured freely by the user.
- The FP gateway's time zone must be set to +0000. Configuration is performed using the `TimeZone` parameter in the USER database.
- The FP gateway's time must be set to UTC, as the Deutsche Telekom portal expects a UTC time stamp. Otherwise, your arrays are displayed with a time shift.

After implementing the MQTT client configuration, the FP gateway must be restarted.
You can monitor the MQTT client's connection status in the process branch:
`[<Get _="/Process/CloudConn/" ver="y" />]`

Result (example):

```
<Get>
  <CloudConn>
    <ConnectionType _="mqtt+ssl:nb-iot.ram.m2m.telekom.com:8883"/>
    <ConnectionState _="1"/>
    <ConnectionStateMsg _="connected"/>
    <LastTimeStamp _="Timestamp"/>
    <ChangeToggle _="Togglebit"/>
  </CloudConn>
</Get>
```

ConnectionType indicates the type of connection (mqtt+ssl), the URL / IP address for the broker that is currently active and the connection status.

`ConnectionState` = 0 means: no connection
`ConnectionState` = 1 Connection is established
`ConnectionStateMsg` = (not connected, connecting, connected)
`LastTimeStamp` = The time stamp indicates when the last MQTT message was sent.
Format: DD.MM.YYYY HH:MM:SS Timezone
`ChangeToggle` = the ToggleBit only changes with each new time stamp (LastTimeStamp)

2.1 Realtime sections

The data points that are to be transmitted can be defined in up to 5 realtime sections.

The cycle for each realtime section must be preset. The cycle time is also used to define whether the data is transmitted cyclically or via a hysteresis function.

e.g. `RTSendTime1, …, RTSendTime5.`
The data from the corresponding sections <RealTime1>..</RealTime1>, etc. then belongs to this.

Example:

```
<CloudConn>
    <RTSendTime1 _="-1" />
    <UpdateRate1 _=20 />
    <Realtime1>
       <Volt_L1_N _="/Process/Modbus/D1/Volt_L1_N" hysteresis="2" />
       .
```

```
      .
   </Realtime1>

   <RTSendTime2 _="10" />
   <Realtime2>
      <Obis_180 _="/Process/Meter/MT174/Obis_180"/>
   </Realtime2>
   .
   .
   <RTSendTime5 _="100" />
   <Realtime5>
      <Temp_Board _="/Process/OneWire/Mainboard/TEMP_Board"/>
   </Realtime5>
</CloudConn>
```

2.2 Sending data via the EventHandler or command

Data points can also be sent via EventHandler or by command:

Command (TICO, ...):

```
 <CloudSendRealtimeData [_="CloudConn[,Realtime_Specifier"]] />
```

or

```
 [<CloudSendRealtimeData />]
```

CloudConn (optional) = Name of the CloudConnector. Values: CloudConn, CloudConn2, CloudConn3. Default: CloudConn

Realtime_Specifier (optional) = name of the realtime section, e.g. Realtime1. Default=Realtime1
If the optional parameter is omitted, all defined sections are sent.

Example 1: Send Realtime1 sections from CloudConn

```
 <CloudSendRealtimeData />
```

Example 2: Send realtime section "Realtime2" from CloudConn2

```
 <CloudSendRealtimeData _="CloudConn2,Realtime2" />
```

Example 3: Send realtime section "Realtime3" from CloudConn

```
 <CloudSendRealtimeData _="CloudConn,Realtime3" />
```

Example 4: Send realtime section "Realtime2" from CloudConn2 via an EventHandler

```
 <CloudSendRealTimeDate_2_2>
    <CloudSendRealtimeData _="CloudConn2,Realtime2" />
 </CloudSendRealTimeDate_2_2>
```

Minute-by-minute execution of the EventHandler via scheduler:

```
 <CloudSendRealTimeDate_2_2_SCH _="CloudSendRealTimeDate_2_2">
      <Minute _="0-59"  />
 </CloudSendRealTimeDate_2_2_SCH>
```

Test via TiXML command with TICO

```
 [<DoOn _="CloudSendRealTimeDate_2_2" ver="v" />]
```

2.3 Hysteresis

For Timer 1 (`RTsendTime1`), there is the option to set whether each data point is to be transmitted depending on a hysteresis setting.

Example: Data point “Temperature”

Start value	= 20.5K
Absolute hysteresis	= 0.5

Result: If the data point’s value changes to 20 or 21, this is transmitted.

Regulations:

- All values are transmitted once after restarting the device.
- If the last value that was transmitted changes by the hysteresis amount, the value is transmitted once again.

The required parameters:

`<RTsendTime1 _="-1" />`

The value `-1` means that data is no longer sent cyclically but using the hysteresis parameters that were set.

`<UpdateRate1 _="PollTime" />`

Poll time specifies the frequency with which the values are requested from the process branch. The poll time is specified in seconds. Value range: 1 - 3600

Attention:

The hysteresis function is currently only available for the first timer (`RTsendTime1 and UpdateRate1`). The hysteresis function can only be used for numeric values.

The hysteresis is defined as a floating comma value in the configuration for each realtime value.
Example:

```
hysteresis="1.5"
```

Definition:

- The comma / decimal point is displayed with "."
- Max. 2 decimal places
- No negative numbers

Example:

```
<RTsendTime1 _="-1" />
<UpdateRate1 _="20" />

<Realtime1>
   <Voltage_L1_N _="/Process/Modbus/D1/Voltage_L1_N" hysteresis="2" />
   <CPULoad _="/Process/MB/CPULoad" hysteresis="8.5" />
   <SupplyTemp _="/Process/PV/SupplyTemp" hysteresis="0.5" />
   <ReturnTemp _="/Process/PV/ReturnTemp" hysteresis="0.8" />
</Realtime1>
```

In the example above, the hysteresis is defined for the individual data points.
If the data point `Voltage_L1_N` changes by +/- 2.0 compared to the last value that was transmitted, this value is transmitted. If the data point `CPULoad` changes by +/- 8.5 compared to the last value that was transmitted, this value is transmitted, etc.

2.4 MQTT connection visual indication

The “Signal” LED indicates the MQTT connection status.
Off = no MQTT connection active
Flashing red = MQTT connection is being established
Illuminated green = MQTT connection is established

2.5 Service routing

The connection to the Telekom Cloud is normally established using the active LAN interface.
The TiXML database `ISP/ISP/OUT` can also be used to establish the MQTT connection via a mobile communications connection or a VPN tunnel:

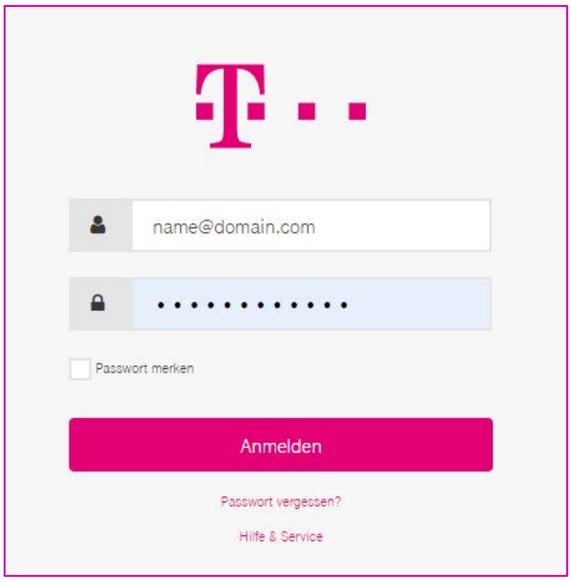
```
[<SetConfig _="ISP/ISP" ver="y">
    <!-- Define communication interface for services -->
    <OUT>
        <SMTP _="MODEM"/>
        <CBIS _="MODEM"/>
        <POP3 _="MODEM"/>
        <URLSend _="MODEM"/>
        <INetTime _="MODEM"/>
        <HTTPConn _="MODEM"/>
        <CloudConn _="Ethernet"/>
        <IBMConn _="MODEM"/>
        <FTPPut _="MODEM"/>
        <SFTPPut _="MODEM"/>
        <VPN _="MODEM"/>
    </OUT>
</SetConfig>]
```

The following options can be used for the MQTT client:

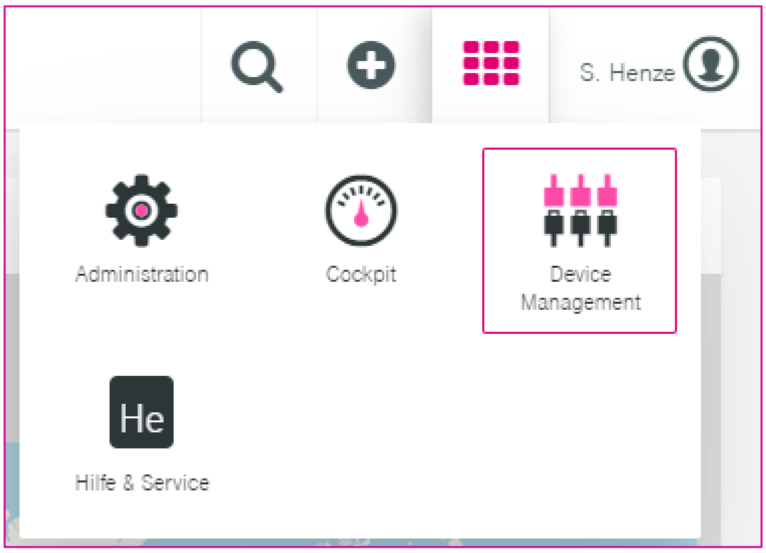
```
Ethernet
MODEM
VPN
```

3 Registering the FP gateway in the Cloud of Things (CoT)

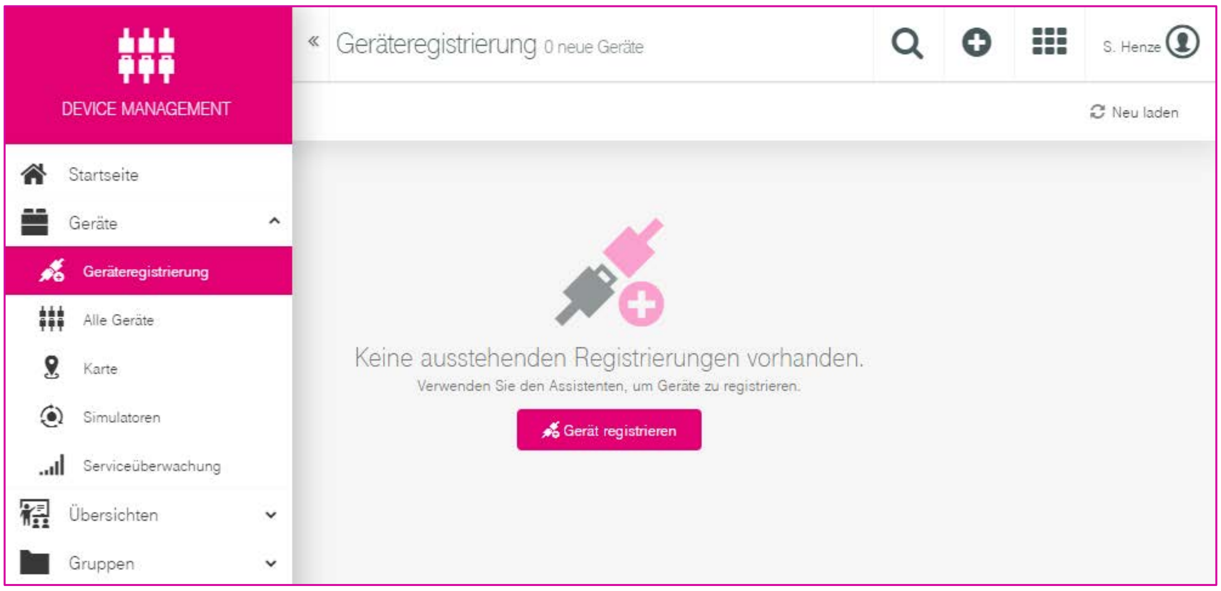
After the configuration was transferred to the device, the FP gateway must be registered in the “Cloud der Dinge” (cloud of things). To do so, use your access data to log in to the “Cloud der Dinge” (cloud of things) portal.



Call device management by clicking the menu icon at the top right and then clicking the “Device Management” menu item:

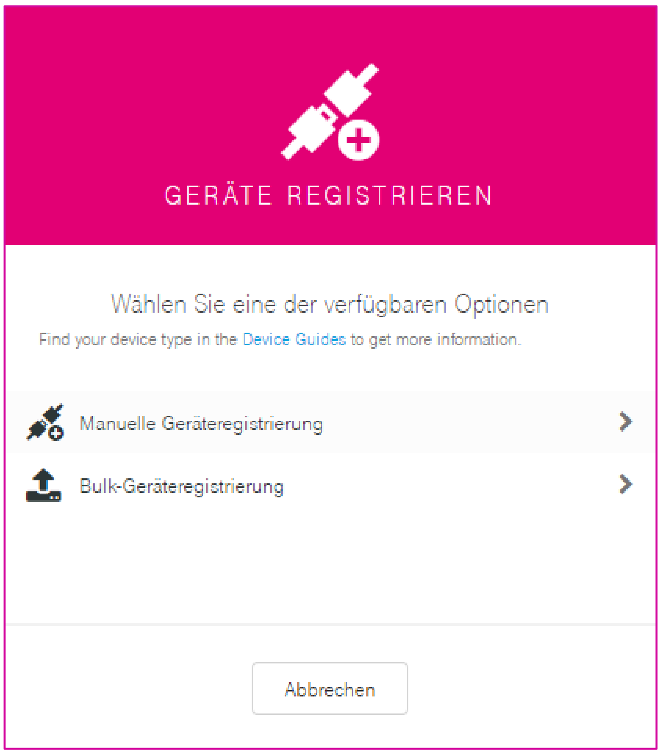


Click “Geräte -> Geräteregistrierung” (Devices -> Device Registration) on the left.



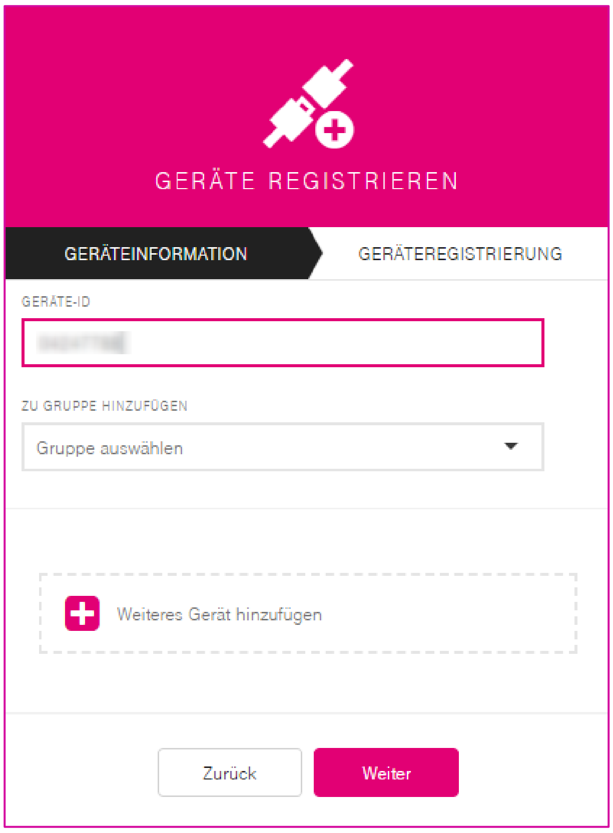
Keine ausstehenden Registrierungen vorhanden.
Verwenden Sie den Assistenten, um Geräte zu registrieren.

Click “Manuelle Geräteregistrierung” (Manual Device Registration).



Wählen Sie eine der verfügbaren Optionen
Find your device type in the Device Guides to get more information.

Then enter your FP gateway’s serial number in the “GERÄTE-ID” (DEVICE ID) field.
The 8-character serial number is located on your gateway’s test label.
The number generally starts with “04”.


GERÄTE REGISTRIEREN

GERÄTEINFORMATION | GERÄTEREGISTRIERUNG

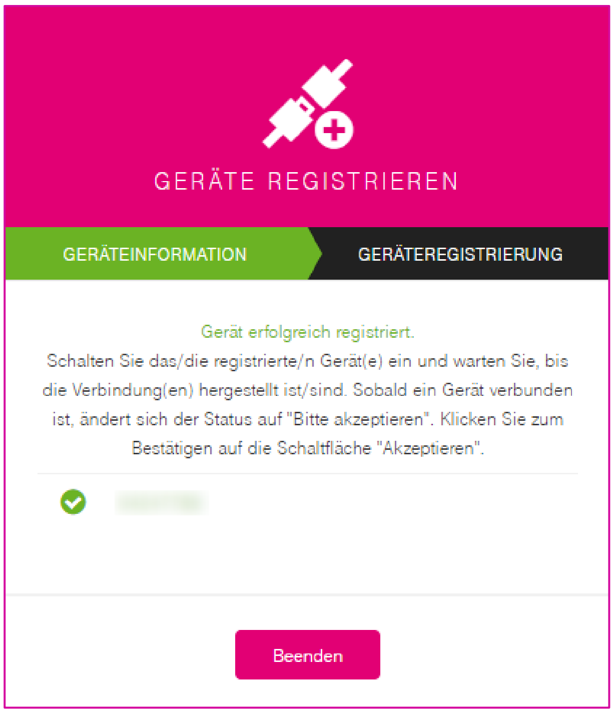
GERÄTE-ID

[redacted]

ZU GRUPPE HINZUFÜGEN

Gruppe auswählen

Then click “Weiter” (Continue).
You receive a message stating that the device was registered successfully:


GERÄTE REGISTRIEREN

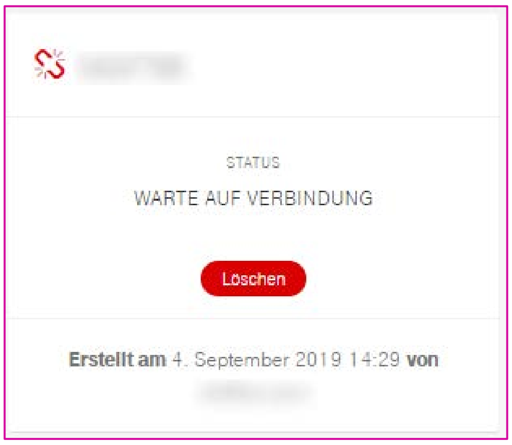
GERÄTEINFORMATION | GERÄTEREGISTRIERUNG

Gerät erfolgreich registriert.
Schalten Sie das/die registrierte/n Gerät(e) ein und warten Sie, bis die Verbindung(en) hergestellt ist/sind. Sobald ein Gerät verbunden ist, ändert sich der Status auf "Bitte akzeptieren". Klicken Sie zum Bestätigen auf die Schaltfläche "Akzeptieren".

✓ [redacted]

Click “Beenden” (Finish).

You will see the following display in the Telekom portal:

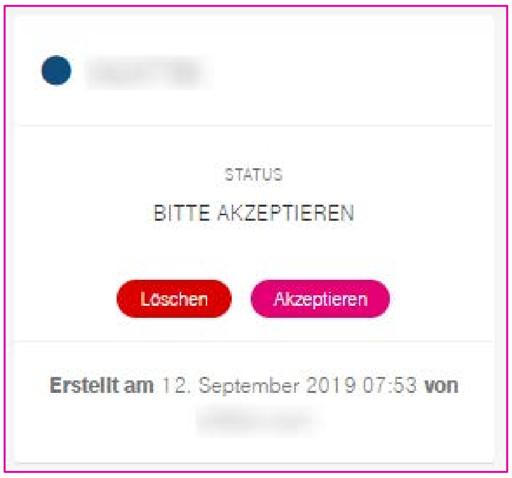

STATUS

WARTE AUF VERBINDUNG

Erstellt am 4. September 2019 14:29 **von**

After approx. 30 - 120 seconds, the “Signal” LED on the FP gateway should flash red.

You will then see an additional “Akzeptieren” (Accept) button in the portal:

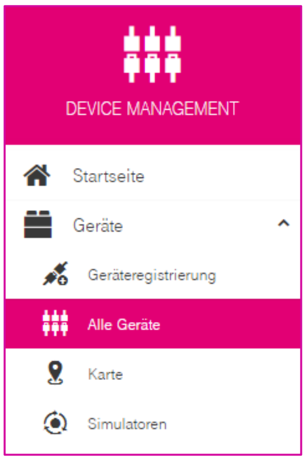

STATUS

BITTE AKZEPTIEREN

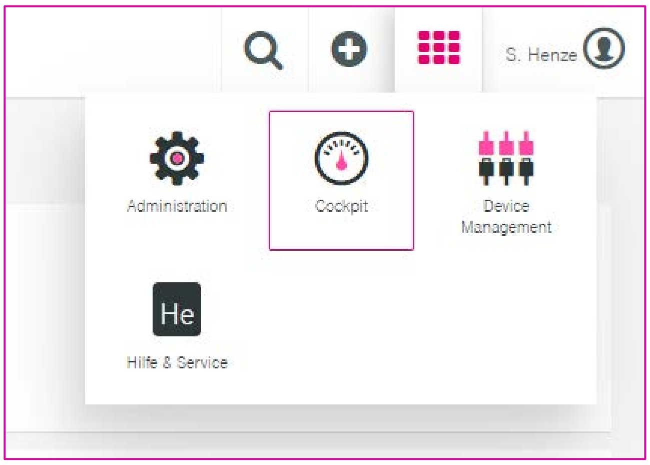
Erstellt am 12. September 2019 07:53 **von**

Now click the “Akzeptieren” (Accept) button.
This completes device registration.

Click the “Alle Geräte” (All Devices) menu item on the left and check whether your device is in the device list.



Now switch to the cockpit view by clicking the following menu item at the top right:



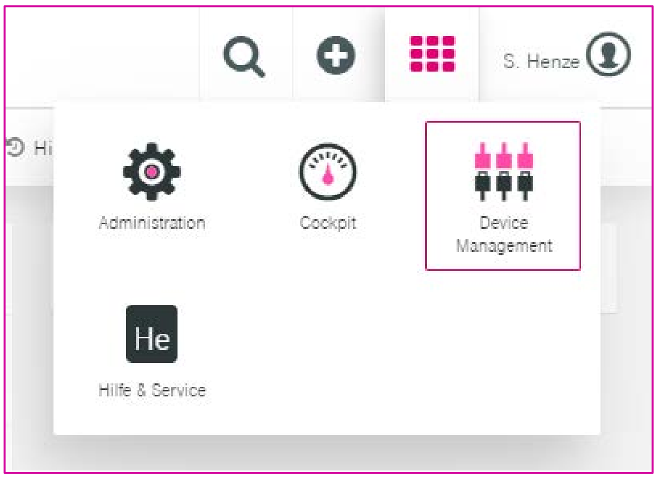
In the cockpit view, call the data explorer and add all data points that are relevant for you.

You can then create your own widgets to display the data.

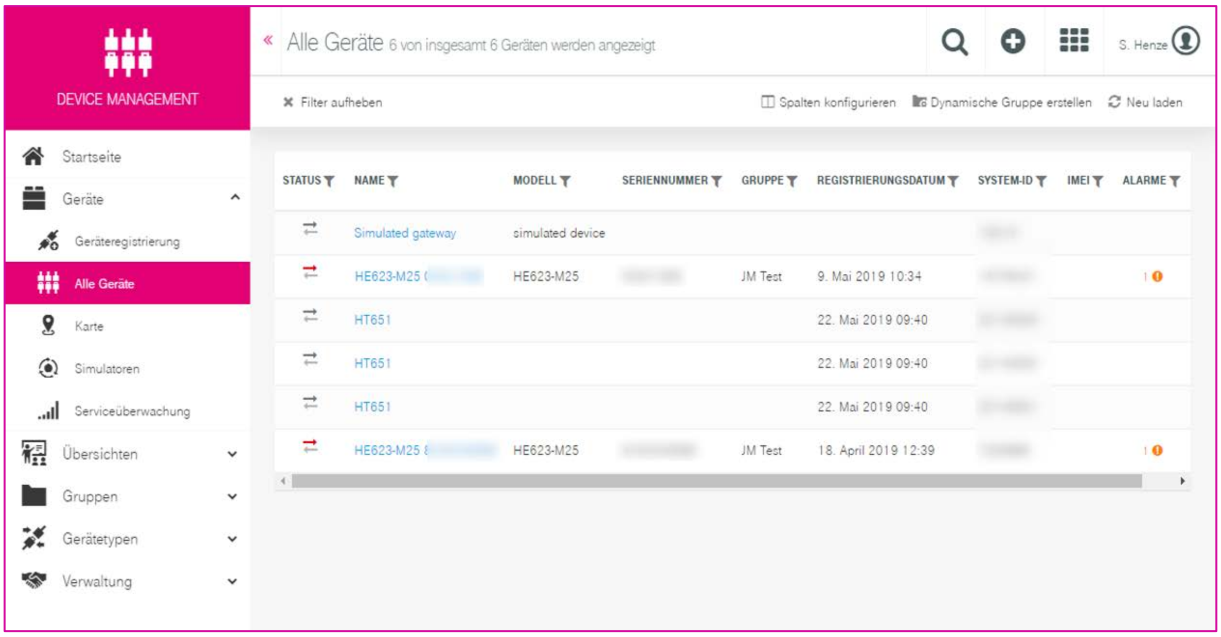
4 Feedback channel to the FP gateway (Shell)

The Telekom broker can use a feedback channel to send TiXML commands to the FP gateway in order to save a new configuration to the device, to restart the device or to switch an output for example. The feedback channel is set up automatically and can be called using "Shell" in the Telekom CoT interface.

In order to use the feedback channel, first switch to "Device Management".



Then click the "Geräte" (Devices) entry in the menu on the left.

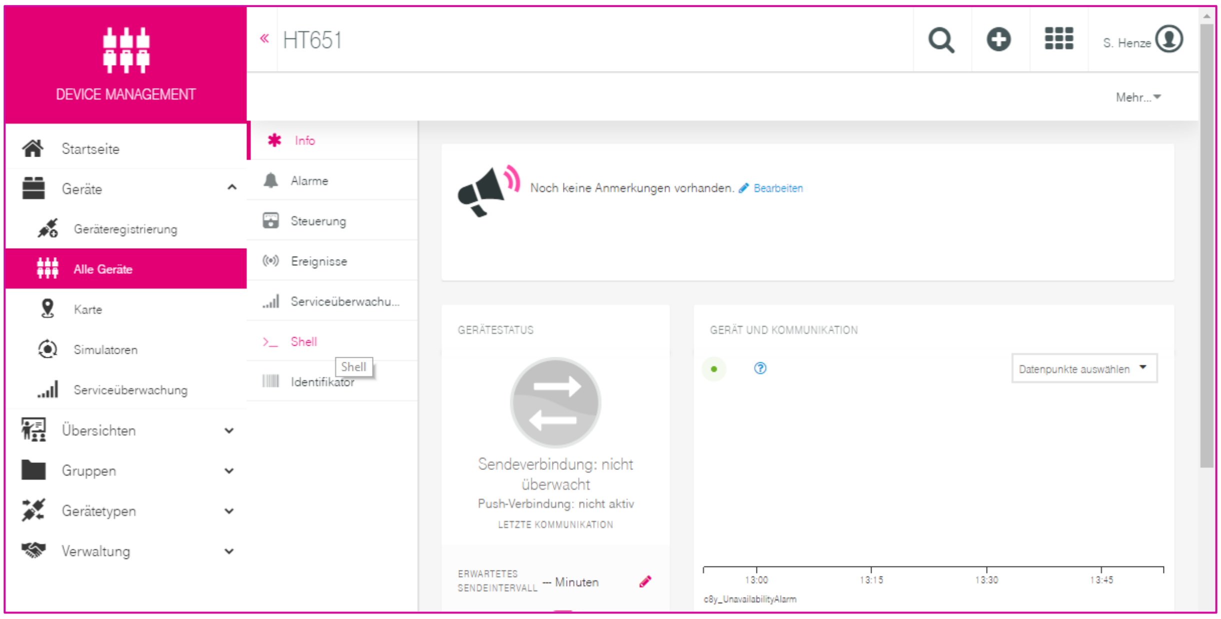


Alle Geräte 6 von insgesamt 6 Geräten werden angezeigt

STATUS	NAME	MODELL	SERIENNUMMER	GRUPPE	REGISTRIERUNGSDATUM	SYSTEM-ID	IMEI	ALARME
⇄	Simulated gateway	simulated device				[redacted]		
⇄	HE623-M25 ([redacted]	HE623-M25	[redacted]	JM Test	9. Mai 2019 10:34	[redacted]		1 ❗
⇄	HT651				22. Mai 2019 09:40	[redacted]		
⇄	HT651				22. Mai 2019 09:40	[redacted]		
⇄	HT651				22. Mai 2019 09:40	[redacted]		
⇄	HE623-M25 [redacted]	HE623-M25	[redacted]	JM Test	18. April 2019 12:39	[redacted]		1 ❗

Now select the required device from the list on the right by clicking the entry in the “Name” column.

Then click the “>_ Shell” entry on the left.



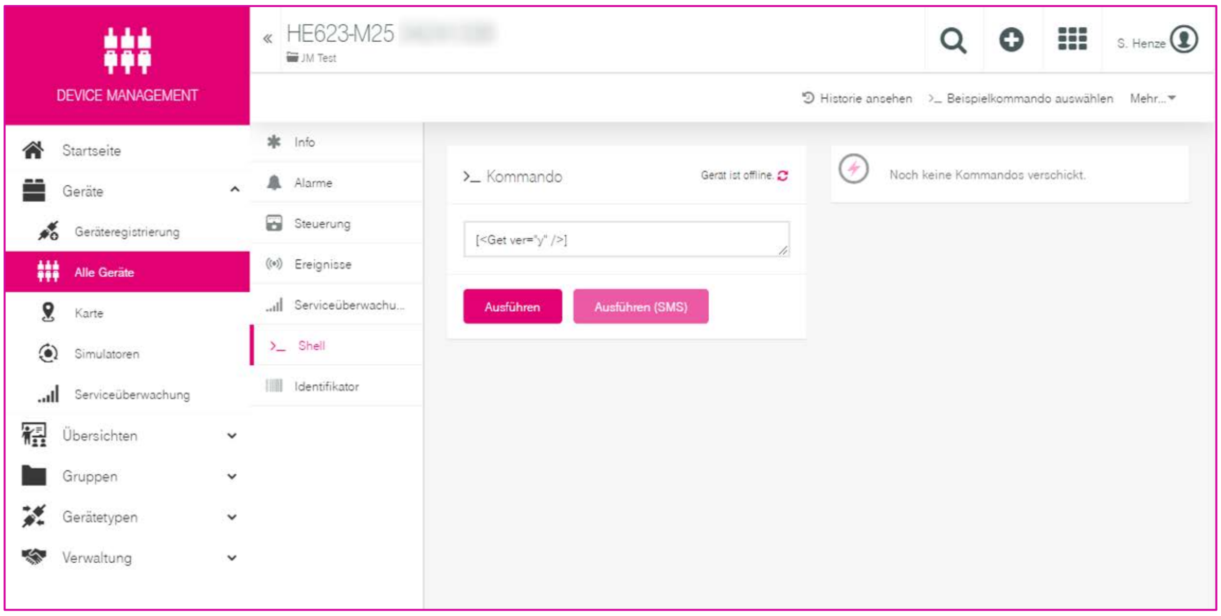
Noch keine Anmerkungen vorhanden.

Sendeverbindung: nicht überwacht

Push-Verbindung: nicht aktiv

ERWARTETES SENDEINTERVALL -- Minuten

In the “” field, you can now enter a valid TiXML command or click the „>_ Beispielkommando …“ option to select a command that was defined previously.



[<Get ver="y" />]

Noch keine Kommandos verschickt.

Click the “Ausführen” (Execute) button to send the command to the gateway. The gateway should respond within a few seconds. The response is shown at the top right and can be expanded.

Important:
Before each configuration command (i.e. all `SetConfig` commands), internal processing should always be stopped using the following command:

```
[<Set _="/Process/Program/Mode" value="Stop" ver="v"/>]
```

Wait for the result of the aforementioned command. A `SetConfig` command should only be executed once the aforementioned stop command has been executed successfully.
If no response to the stop command is received within 15 seconds, the stop command must be sent again.
After the configuration database(s) was/were sent to the device, the following command can be used to restart process editing:

```
[<Set _="/Process/Program/Mode" value="Run" ver="v"/>]
```

4.1 Gateway error messages

If the gateway receives commands from the broker and executes these commands, error messages are sometimes generated if the command contains incorrect parameters for example, or cannot be executed for other reasons.

A parameter can be used when sending the command to specify how comprehensive the response to the gateway error messages is.

There are three possible formats that can be used to transmit these errors. A short or a long error message is issued depending on the specification by the “ver” parameter See the table below.

Command example:

```
    [<Get _="/Processs/" ver="vmode"/>]
```

vmode	Description	Example
n	Error message as numeric value	`<Error _="-2196" />`

y	Error message, short description	`<Error>`
		`  <ErrNo _="-2196" />`
		`  <ErrText _="path to key not found" />`
		`  <ErrorCause>`
		`    <ErrNo _="-2196" />`
		`    <ErrText _="path to key not found" />`
		`    <Class _="TXSTCPGetSetValueCmd" />`
		`  </ErrorCause>`
		`</Error>`

v	Error message, long description	`<Error>`
		`  <ErrNo _="-2196" />`
		`  <ErrText _="path to key not found" />`
		`  <ErrorCause>`
		`    <ErrNo _="-2196" />`
		`    <ErrText _="path to key not found" />`
		`    <Line _="127" />`
		`    <Module _="SSet.cpp" />`
		`    <Class _="TXSTCPGetSetValueCmd" />`
		`  </ErrorCause>`
		`</Error>`

5 Alert function

5.1 Overview

The FP gateways can be configured via TiXML so that alarms can be triggered directly in the cloud. These alarms can then be processed further in the cloud, e.g. using a smart rule.

The following data is transmitted to the cloud in the event of an alarm:

`type`:	Alarm type as text (freely configurable in the gateway); max. 50 characters
`datetime`:	Alarm date and time (format: ISO 8601) and time zone
`text`:	Descriptive text (freely configurable); max. 50 characters
`status`:	Alarm status. Values: `1` (=`ACTIVE`) or `0` (=`CLEARED`); 1 character
`severity`:	Severity. `CRITICAL`, `MAJOR`, `MINOR`, `WARNING`

The configuration can be used to generate alarms for an entire group of data points or other system properties up to each individual data point. Alarms are detected and triggered in the exact same way as local alarms (SMS, e-mail) on the gateway. The alarm itself is then triggered by the `SendAlarmECN` EventHandler command. In addition, the FP gateway's alarm handler sends a sequential ID for each alarm.

The alarms sent are also logged in the "Event" system database. Example:

```
<ID_31 _="2020/08/14,09:18:58">
   <Event>
      <Event _="ECN_Alarm" Origin="EventState">
         <AlarmCode _="ServiceButton" />
         <AlarmText _="Service button alarm triggered" />
         <State _="1" />
         <TimeStamp _="2020/08/14 09:18:58 +0000" />
         <Severity _="WARNING" />
      </Event>
   </Event>
</ID_31>
```

5.2 Setting TiXML parameters

Basic principle for setting TiXML parameters

`EventState` -> triggers `EventHandler` -> calls `SendAlarmECN`.

`SendAlarmECN` uses `MessageJobTemplates` and `MessageText` definitions and uses `ProcessVars` and different SystemProperties if required.

The basic mechanisms for configuring the aforementioned databases are provided in the TiXML Reference Manual.

Sample configuration

In the following example, the service button is used to trigger a cloud alarm. The "`PollButton_Alarm`" EventState is used to trigger the "`ECN_Alarm`" EventHandler when the service button is pressed or released again. Five additional parameters are transferred to the EventHandler.

14-EventStates.txt

An `EventState` is triggered as soon as the user presses the service button.
In this case, the "`ECN_Alarm`" `EventHandler` is called with additional parameters (`AlarmCode, AlarmText, State, TimeStamp,` and `Severity`).
The parameters are partly determined via references, e.g. the service button's status is transferred as an AlarmState (1=pressed; 0=released). The time is read out from the `TIMES/DATE` and `TIMES/TIME` SystemProperty.

```
[<SetConfig _="PROCCFG" ver="y">
<EventStates>
<!-- ALARM Emergency-Off -->
 <PollButton_Alarm>
   <Enabled _="TRUE" />
   <ProcessVar _="/Process/MB/PollButton" flank="both" />
     <Event _="ECN_Alarm" >
      <AlarmCode _="ServiceButton" />
      <AlarmText _="Service button alarm triggered" />
      <State _="&#xae;/Process/MB/PollButton;" />
      <TimeStamp _="&#xae;/TIMES/DATE; &#xae;/TIMES/TIME; &#xae;/USER/USER/TimeZone;" />
      <Severity _="WARNING" />
     </Event>
 </PollButton_Alarm>
</EventStates>
</SetConfig>]
```

A maximum total of 100 different EventStates can be configured.

10-EventHandler.txt

The `EventHandler` configuration (generic, can be used for all cloud alarms) is defined as follows:

```
[<SetConfig _="TEMPLATE" ver="y">
<EventHandler>
  <ECN_Alarm>
    <!-- compute new Alarm_Id -->
    <Process _="Counter"/>
    <!-- now send alarm using cloudConnector to cloud -->
    <SendAlarmECN _="MessageJobTemplates/Alarm_ECN" />
    <!-- compute new Alarm_Id -->
    <Process _="Counter"/>
  </ECN_Alarm>
</EventHandler>
</SetConfig>]
```

11-MessageJobTemplates.txt

The `MessageJobTemplate` configuration (generic, can be used for all cloud alarms) is defined as follows:

```
[<SetConfig _="TEMPLATE" ver="y">
<MessageJobTemplates>
  <!-- send alarm to cloud -->
  <Alarm_ECN _="WriteFile">
     <Body _="/D/UserTemplates/ECN_AlarmText"/>
  </Alarm_ECN>
</MessageJobTemplates>
</SetConfig>]
```

12-MessageText.txt

The `MessageText` configuration (generic, can be used for all cloud alarms) is defined as follows:

```
[<SetConfig _="TEMPLATE" ver="y">
<!-- Definition of UserTemplates database -->
<UserTemplates>
  <LocationText hidden="1">
    <Email>
      <C _=""/>
    </Email>
    <SMS _=""/>
  </LocationText>
  <Attachments hidden="1"/>
  <!-- alarm text for cloud connector in xml format -->
  <ECN_AlarmText>
     <S _="&lt;alarm id=&quot;&#xae;/Process/PV/Alarm_Id;&quot; " />
     <S _="type=&quot;&#xae;~/AlarmCode;&quot; " />
     <S _="datetime=&quot;&#xae;~/TimeStamp;&quot; " />
     <S _="errorcode=&quot;&#xae;~/AlarmText;&quot; " />
     <S _="state=&quot;&#xae;~/State;&quot; " />
     <S _="severity=&quot;&#xae;~/Severity;&quot; " />
     <S _="/&gt;" />
  </ECN_AlarmText>
</UserTemplates>
</SetConfig>]
```

13-ProcessVars

At least 2 process variables (PV) must be defined in the `ProcessVars` configuration: `Alarm_Id` and `Counter`:

```
[<SetConfig _="PROCCFG" ver="y">
<ProcessVars>
<!-- Alarm ID -->
<Alarm_Id def="0" />
<!-- Alarm_Id will be increased automatically via event handler -->
<Counter>
   <Process>
       <LD _="/Process/PV/Alarm_Id" />
       <ADD _= "1" />
       <ST _="/Process/PV/Alarm_Id" />
   </Process>
</Counter>
</ProcessVars>
</SetConfig>]
```

Please note:

Any digital input, PLC variable (external) or process variable (PV) can be used to trigger the alarm.