MQTT Proxy

Installation Manual

Table of contents

List of Figures

List of Tables

1. MQTT Proxy Features

Cascoda's MQTT proxy enables forwarding of KNX IOT messages to MQTT. It is part of the KNX IoT Hub as an add on feature.

The MQTT proxy can forward the following data formats:

- Temperature
- Relative Humidity
- Atmospheric Pressure
- Carbon Dioxide (CO2)
- Volatile Organic Compound (VOC) Index
- Battery Status
- On/Off

2. General information

2.1. Document Version information

This manual is amended periodically and will be brought into line with new software releases. The change status (date) can be found in the contents header. If you have a device with a later software version, please check www.cascoda.com to find out whether a more up-to date version of the manual is available.

2.2. Used Terms

Sign	Description
DANGER!	Indicates an immediately hazardous situation which will lead to death or severe injuries if it is not avoided.
CAUTION!	Indicates a potentially hazardous situation which may lead to trivial or minor injuries if it is not avoided.
WARNING!	Indicates a situation which may lead to damage to property if it is not avoided.
NOTE!	Indicates a situation which may lead to possible (known) side effects.

Table 1: *Used Terms*

2.3. Safety instructions

Not applicable.

2.4. Issues

Questions about the product?

You can reach the technical service of Cascoda under Tel. +44 (0)2380 638 111 or support@cascoda.com.

We need the following information to process your service request:

- Type of appliance (model name or item number)
- Description of the problem
- Serial number or software version
- Source of supply (dealer/installer who bought the device from Cascoda)

For questions about KNX functions:

- Version of the device application
- ETS version used for the project

2.5. Contact information

info@cascoda.com
Threefield House,
Threefield Lane,
Southampton,
SO14 3LP, UK

3. Configuration

3.1. Device Startup

The device is start up when the KNX IoT Hub is powered on.

3.2. Commissioning

Configuration is made using the KNX software as of ETS 6.3 or later. The product file can be downloaded from the ETS online catalogue and the Cascoda website.

The KNX IoT MQTT proxy page can be found at the KNX IoT Tab.

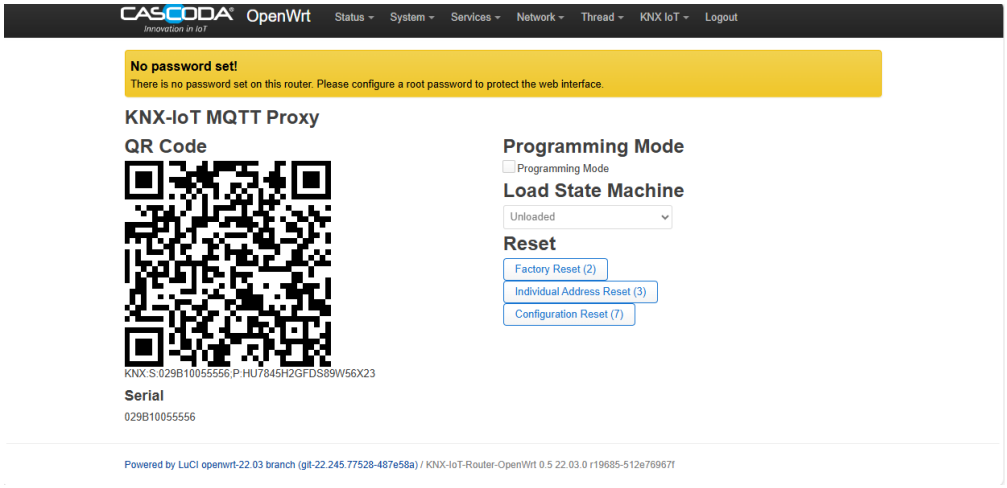

Fig 1: MQTT proxy page

This page shows the QR code which can be used by ETS.

The QR code can be used to for KNX commissioning.

Check out the youtube video here, demonstrating the process of doing (Thread and) KNX commissioning using a QR code scanner.

3.2.1. KNX Commissioning

KNX commissioning is adding the device to an ETS project. Since KNX IoT is a secure KNX protocol, one needs to have the security credentials and the serial number of the device. This information is contained in the QR code.

The device can only be added to an KNX IoT Area or Line. When the device is added to a KNX IoT area or Line, the credentials can be supplied. ETS can scan the QR code with the camera (or 2D bar code scanner).

3.2.1.1. Adding the Device to a Topology

1. Open the project provided in ETS version 6.3 or later.
2. Open the "Devices" panel, and select the device that you will use for this demo. In the "Properties" section on the right, click "Add Device Certificate". A window will pop up, with an input field for the QR code information. Use a QR code scanner to scan the QR code provided in this manual. (There is also an option to use a camera).

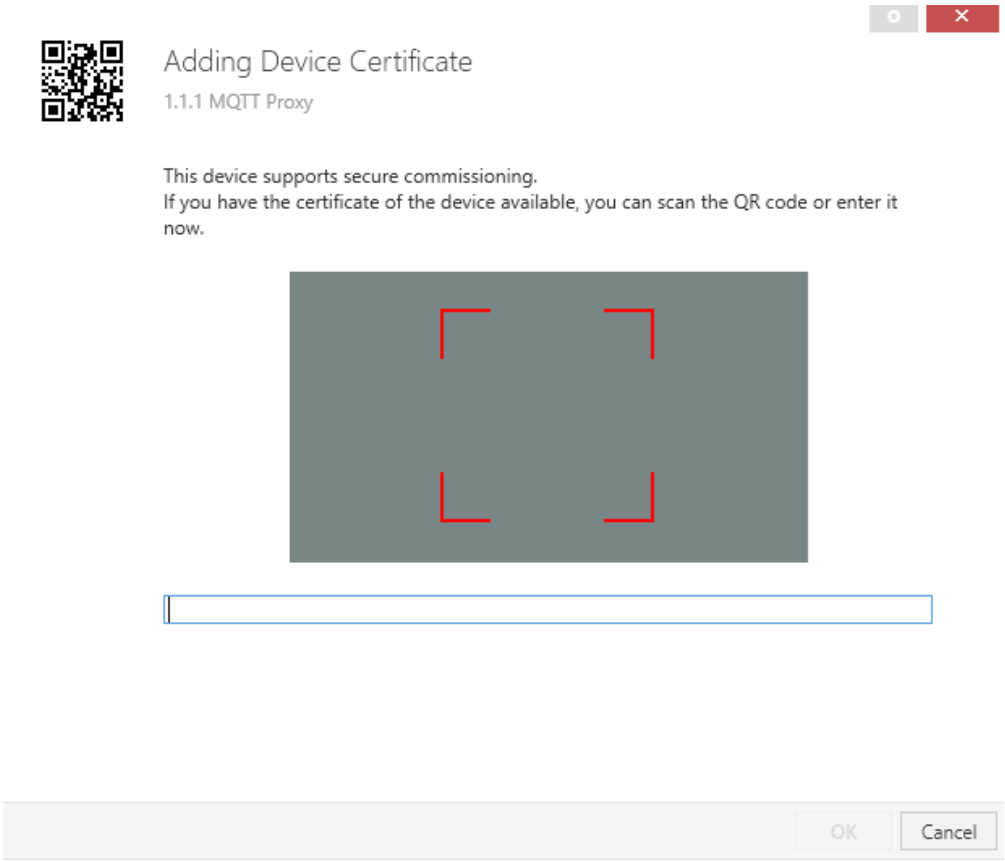

Fig 2: Add Certificate

3. Open a "Topology" panel. Drag and drop the device into an IoT area.

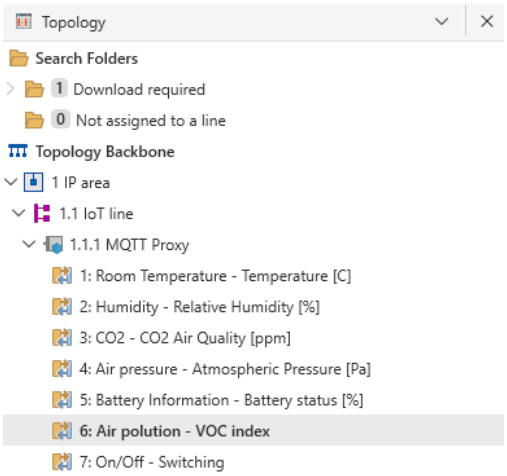

Fig 3: ETS Topology

3.2.1.2. Creating a Configuration

These steps create the group objects that will be used in the s-mode messages. By linking the group objects to the communication objects (data points), one can send s-mode messages to actuators, and receive s-mode messages from sensors.

1. Open a "Group Addresses" panel.

2. Select "Group Addresses", right click, select "Add Main Groups", click OK.

3. Select the newly created main group, right click, select "Add Middle Groups", click OK.

4. Select the newly created middle group, right click, select "Add Group Addresses", increase the count to however many group addresses you want (e.g. the same as the number of Group Objects that you would like to link), then press "OK".

5. Now open the "Group Objects" tab for your device, and link each group object that you want by right clicking, "Link with...", and then selecting the group address.

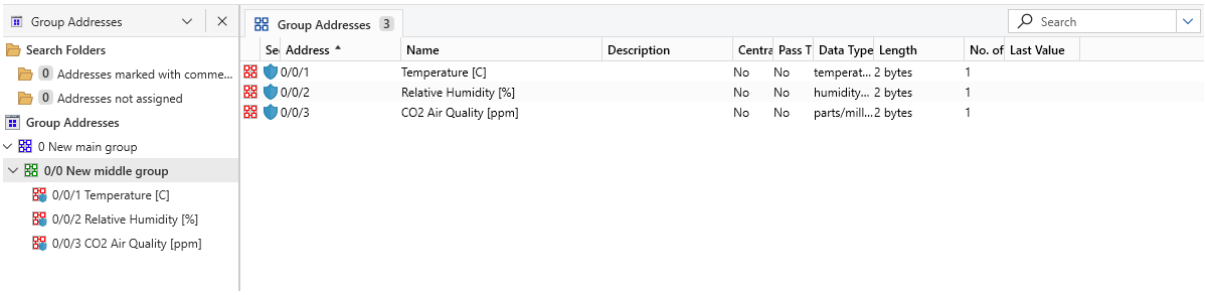

Fig 4: ETS Topology

3.2.1.3. Downloading the ETS configuration

The downloading of the configuration can happen when the ETS data for the data is created, e.g.:

- The parameters are set
- The communication objects are linked with group.

If no communication objects are linked with group then no MQTT messages are send.

The download can be started in ETS, and one can use either:

- Download by serial number
- Download per programming mode

The download by serial number does not require any interaction with the device. All you have to do is click on the Use Serial Number button.

The download per programming button requires pressing the **Programming mode** toggle, e.g. the toggle is turned on. Disabling the programming mode can achieved by turning off the **Programming mode** toggle.

Once the download is complete, the device becomes fully operational and functional.

3.2.1.3.1. Troubleshooting failed downloads

If you are unable to complete a download with ETS, please first check if you've entered the correct details for the MQTT broker; If the broker is an online one, please also check if the Hub is connected to the internet.

Failed downloads can also be caused by the IPv6 addresses allocated to the Hub by the upstream router. Please attempt the following steps:

1. Navigate to Network -> Interfaces.
2. Find the WAN6 Interface within the list, and press the corresponding "Edit" button.
3. Navigate to the Advanced Settings tab, uncheck "Delegate IPv6 prefixes" and set "IPv6 assignment length" to "disabled".
4. Press Save, then press Save and Apply
5. Restart the KNX IoT Hub and ETS.

If you are still having difficulties completing a download, you can additionally unplug any upstream Ethernet cables connected to the WAN interface, to ensure no global IPv6 addresses are allocated to the Hub.

3.2.1.4. Checking MQTT connectivity status

The MQTT status is being logged. The status logs can be found at:

Menu Bar -> **Status** -> **System Logs**

Select: **System Log**

Message from the MQTT proxy are prefixed with knx_mqtt_logger

Example:

```
Wed Aug  7 09:47:33 2024 daemon.info knx_mqtt_logger[1991]: mqtt_init...
Wed Aug  7 09:47:33 2024 daemon.info knx_mqtt_logger[1991]: MQTT configuration:
Wed Aug  7 09:47:33 2024 daemon.info knx_mqtt_logger[1991]:  hostname: c11217c0b8ae45e2ac217e6aa659fa49.s1.eu.h
..
```

3.2.1.5. Reset of the MQTT proxy

- Reset KNX

Reset of KNX is achieve by pressing the **Factory Reset** button.

> **NOTE!** KNX Reset: this means that also the security credentials are removed.
> Hence ETS will download newly created device keys.

3.3. MQTT

3.3.1. Introduction

MQTT is a publisher subscribe mechanism. E.g. a central MQTT server connects publishers and subscribers. The publisher publishes *payload* (data) on a *topic* and the *payload* format is JSON. The subscriber can receive *topics* by subscribing them via the MQTT Server.

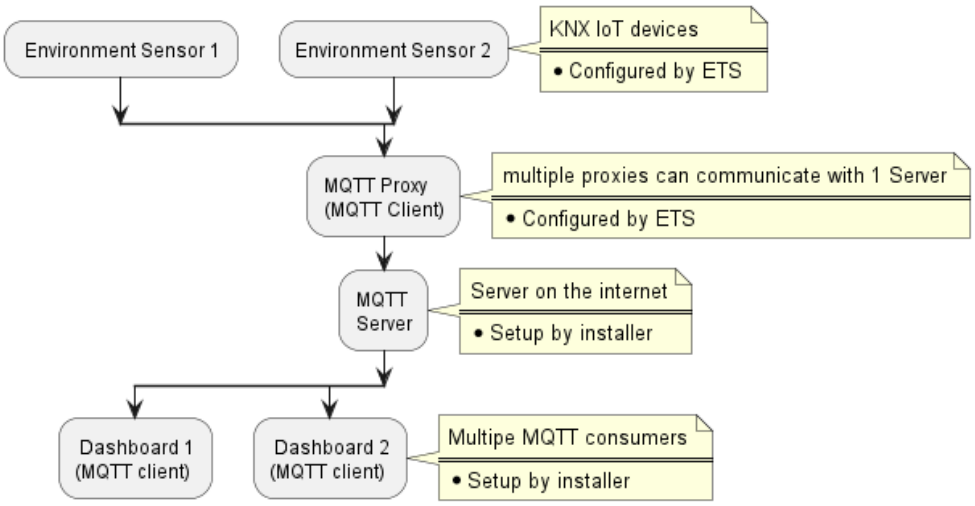

Fig 5: The MQTT proxy as part of a system.

In the above image the **Environment Sensors** are publishing their data and the **dashboards** are consuming the data. Hence the **dashboards** are subscribing to the *topics* that are used by the **Environment Sensors**. Hence these *topics* and the structure of the payloads needs to be known, and are described in this document.

3.3.2. GA (s-mode) topic structure

Used for transporting the s-mode message per group *topic* can be filtered by:

- IID (Installation IDentifier).
- IA (Individual Address, e.g. who sends it).
- GA (Group Address, e.g. which value is being send).

Notation:
KNX/[IID]/[IA]/[GA]
Example:
KNX/245634/23/5

- The Installation ID might have different values, depending on how many proxies are used.
- The Individual Address is from a specific KNX device.
- The Group Address is a specific measurement of a data point inside a device.

Typical implementation would subscribe to

KNX/#

so that it will receive all data of all **mqtt proxies** connected to the MQTT Server.

3.3.2.1. Payload

The conversion is using the straight forward conversion from CBOR to JSON, replacing the number with numbers surrounded with quotes.

- s-mode *payload* in json with numbers as tags

```
{
  "4" : 3,              - sia (sender individual address)
  "5" : {
    "7" : 1510,         - group address
    "6" : "w",          - [w]rite, [r]ead, [a]nswer
    "1" : 12.000000     - value of a data point
  }
}
```

3.3.3. meta data topic structure

This *topic* is used for transporting the meta data of a device (proxy) on a group address *topic* can be filtered by:

- IID (installation identifier)
- GA (group address)

Notation:
KNXM/[IID]/[IA]/[GA]
Example:
KNXM/245634/23/5

The **mqtt proxy** transverses the group object table and send out the meta data of the data point that is linked.

The meta data message is be send with "retained" option. This is needed so that when ever a new consumer is started, the consumer will receive the meta data of the installation.

Typical implementation would subscribe to:

KNXM/#

so that it will receive all meta data of all **mqtt proxies** connected to the MQTT Server.

3.3.3.1. meta data payload

- tag structure (json)

```
{
  "url" : "/p/xxx",                  // the url of the data point
  "desc" : "mydescription",          // description of what the url contains
```

```
  "if" : "if.a",                    // interface actuator(input) or sensor(output)
  "unit": "V",                      // unit conveyed
  "IID name" : "my installation name", // user readable name of the IID
  "IA1" : "0/0/1 device name 1",      // user readable name of the device IA
  "IA2" : "0/0/2 device name 2",
  ...
  "IA10" : "0/0/10 device name 10",
  ..
}
```

Note that IA1 .. IA10 can be set by ETS.

3.3.3.1.1. Setting a name for the Individual Address

The property page has the option to enter 10 mapping for the individual address to a name. The **Nr Individual Address to Name mappings** parameter is a drop down to select the amount of name mappings. When **Nr Individual Address to Name mappings** is larger than 0, one can enter the Individual Address (IA) and the name for the device for each appearing entry. The syntax is the individual address in 3 level format, as used in ETS, followed by a space and the name to be used. Note that the syntax is enforced by a regular expression.

3.3.4. network data topic structure

This *topic* is used for transporting the network data of a device (proxy) on a group address. *topic* can be filtered by:

- IID (installation identifier)
- IA (proxy individual address)
- DATA identifier:
- 0 : network diagram info
- 1 : network neighbor information (of the Hub)
- 2 : knx-iot mdns data (as seen by the Hub)
- 3 : rssi mdns data (as seen by the Hub)

Notation:
KNXN/[IID]/[IA]/[DATA identifier]
Example:
KNXN/245634/23/0

The network data is sent regularly, alternating between network diagram info, the network neighbor info and mdns info. The network data topics are retained.

3.3.4.1. network diagram information

The data returned is a JSON document:

```
{
   "networkdata0": {
          "rloc": "0x1000",
          "routedata": [
                {
                      "routerid": 41,
```



```
                "rloc": "0xa400"
            }
        ],
        "childdata": [

        ]
    },
    "networkdata1": {
        "rloc": "0xa400",
        "routedata": [
            {
                "routerid": 4,
                "rloc": "0x1000"
            }
        ],
        "childdata": [
            {
                "rloc": "0xa401",
                "mode": 0
            }
        ]
    }
}
```


Description:

- networkdataX : description of a router
- rloc : the rloc of the router
- routedata : list of router connections
  - rloc : connection to another router
- childdata : list of children of the router
  - rloc : connection to the child

Explanation:

- Router 0xa400 has child 0x4a01
- Router 0xa400 has router connection to 0x1000
- Router 0x1000 has router connection to ox4000

3.3.4.2. network neighbor information

The network neighbor information is only describing the neighbors of the Hub.

The data returned is a JSON document:


```
{
  "neighbor_list":
    [
      {
        "Role":"C",
        "Rloc16":"0x1401",
        "Age":"29",
        "AvgRssi":"-82",
        "LastRssi":"-81",
        "Mode":"",
        "ExtAddress":"422d89c64e63c142",
        "LinkQualityIn":3
```



```
      }
    ],
    "Error":0
}
```

Description:

- neighbor_list : array of neighbors, containing:
- Role : ["C","R","L"]
    - C, child node
    - R, router node
    - L, leader node
- Rloc16 : Routing locator
- Age : time elapsed since the Hub has last received a message from this device
- AvgRssi : average RSSI
- LastRssi : RSSI determined with the AGE
- Mode : ["r","d","n"]
    - r: rx-on-when-idle
    - d: Full Thread Device
    - n: Full Network Data
- ExtAddress : the IEEE 802.15.4 Extended Address.
- LinkQualityIn : The link quality; [0,3], 3 is best.

3.3.4.3. knx-iot mdns information

The relevant knx mdns data is collected by the Hub and sent on the topic. This data can be used to relate the thread information with KNX information. The data is collected by avahi with the following command:

```
avahi-browse -trp _cascoda-knx-iot._udp
```

example result (filtered on wpan0 interface):

```
=;wpan0;IPv6;029b10010710;_cascoda-knx-iot._udp;local;knx-029b10010710.local;fd36:<>:ba00;51805;"rloc16=1006" "exta
```

Note that the ipv6 address is truncated for display purposes

The text fields contain:

- rloc16: The Thread Routing locator
- extadd: The Thread IEEE 802.15.4 Extended Address.
- iid: KNX installation identifier
- ia: KNX individual address (in Hex)

The KNX serial number is the 3rd field. example: 029b10010710

3.3.4.4. RSSI mdns information

The relevant RSSI data is collected by the Hub and sent on the topic. This data can be used to relate the Thread information with additional RSSI information. The RSSI information is only sent out by a Thread router or leader node. Thread child nodes will NOT send out RSSI information. The RSSI information is collected by a Thread router or leader and sent out periodically. When the device is in programming mode, the data is collected every minute, when not in programming mode the data is collected every hour. The data collected is: - RSSI value - RLOC16 of the receiving device (aka router is the receiving side) - RLOC16 of the sending device (the child/router/leader sending the message) - Link quality - Carrier sense Only 1 value is being sent by the router/leader, i.e. it determines the worst RSSI value of all its neighbors. The data is collected by avahi with the following command:

```
avahi-browse -trp _cascoda-r._udp
```

example result (filtered on wpan0 interface):

```
=;wpan0;IPv6;029b10010710;_cascoda-r._udp;local;knx-029b10010710.local;fd36:<>:ba00;51805;"r=1400" "s=1401" "l=3" "
```

Note that the ipv6 address is truncated for display purposes

The text fields contain:

- r: The Thread Routing locator for the receiver e.g. router/leader (in Hex)
- s: The Thread Routing locator for the source (e.g neighbor of r) (in Hex)
- l: Link quality
- c: Carrier sense
- v: RSSI value (positive value)

The KNX serial number is the 3rd field. example: 029b10010710

3.3.5. Management topics (Send)

This *topic* is used to send commands to the hub via MQTT:

- IID (installation identifier)
- IA (proxy individual address)
- DATA identifier:
- 0 : always 0

Notation:
KNXMS/[IID]/[IA]/[DATA identifier]
Example:
KNXMS/245634/23/0

3.3.5.1. Payload information

The data sent is a JSON document. and it can contain the following tag names:

- RefreshMetaData, boolean, refresh MetaData for this Hub
- value is optional

- RefreshNetworkData, boolean, refresh Network data for this Hub
- value is optional
- ChangeNWRefreshTimeOut, integer, change the network refresh time out.
- value is optional
- this value is normally set to 1 minute, however when the installation is complete this value can be set to another value.
- value not allowed: 0
- typical value = 60, i.e. 1 hour.
- this value is persistent over reboots.

JSON example of payload:

```
{
  "RefreshMetaData": true,
  "RefreshNetworkData": true,
  "ChangeNWRefreshTimeOut": 60
}
```

4. Software Bill of Materials

This paragraph contains the list of used open source software in this product.

Name	Version	License
Paho C	1.3.13	Eclipse Public License - v 2.0

Table 2: *Software Bill of Materials*

4.1. Paho C

- Description: C client for MQTT
- License: Eclipse Public License - v 2.0
- Version: 1.3.13
- URL: https://github.com/eclipse/paho.mqtt.c
- Notes: using ssl (secure) version

5. KNX device information

Info Field	Value
Manufacturer	cascoda
Model	mqtt proxy
Order_number	0008
Hardware_type	000000000008
Hardware version	[0, 1, 2]
Firmware version	[0, 0, 1]
Sleepy Device	No

5.1. Data points

url	name	instance	resource type	interface type	data type
"/p/1"	Temperature	1	397.61	if.a	DPT_Value_Temp
"/p/2"	Humidity	1	50002.3075	if.a	DPT_Value_Humidity
"/p/3"	CO2 AirQuality	1	50002.3076	if.a	DPT_Value_AirQuality
"/p/4"	Pressure	1	50002.3077	if.a	DPT_Value_Pres
"/p/5"	Battery Information	1	50002.3078	if.a	DPT_Battery_Info
"/p/6"	VOC index	1	50002.3079	if.a	DPT_Value_VOC_Index
"/p/7"	LED_1	1	417.52	if.a	DPT_Switch

Table 3: *Data points*

5.2. Parameters

url	name	param type
"/p/p5"	IID name	DPT_VarString_8859_1
"/p/p1"	MQTT hostname	DPT_VarString_8859_1
"/p/p2"	MQTT port number	int
"/p/p3"	MQTT username	DPT_VarString_8859_1
"/p/p4"	MQTT password	DPT_VarString_8859_1
"/p/p11"	IA name 1	DPT_VarString_8859_1
"/p/p12"	IA name 2	DPT_VarString_8859_1
"/p/p13"	IA name 3	DPT_VarString_8859_1
"/p/p14"	IA name 4	DPT_VarString_8859_1
"/p/p15"	IA name 5	DPT_VarString_8859_1
"/p/p16"	IA name 6	DPT_VarString_8859_1
"/p/p17"	IA name 7	DPT_VarString_8859_1
"/p/p18"	IA name 8	DPT_VarString_8859_1
"/p/p19"	IA name 9	DPT_VarString_8859_1
"/p/p20"	IA name 10	DPT_VarString_8859_1

Table 4: *Parameters*

5.2.1. Parameter IID name

name of the installation, translating the IID to this name.
Example: my installation name

5.2.2. Parameter MQTT hostname

The MQTT host name.
Only a TLS secured MQTT server can be used
Example: localhost

5.2.3. Parameter MQTT port number

The MQTT port number

Example: 8883

used data range: [50, 30000]

5.2.4. Parameter MQTT username

The user name to use to connect to the MQTT server
Example: TestUser

5.2.5. Parameter MQTT password

The password (for the username) to use to connect to the MQTT server
Example: ca5c0da

5.2.6. Parameter IA name 1

IA and name of the individual address, translating the IA to this name.
Example: 0/0/1 my device name 1
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"

5.2.7. Parameter IA name 2

IA and name of the individual address, translating the IA to this name.
Example: 0/0/2 my device name 2
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"

5.2.8. Parameter IA name 3

IA and name of the individual address, translating the IA to this name.
Example: 0/0/3 my device name 3
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"

5.2.9. Parameter IA name 4

IA and name of the individual address, translating the IA to this name.
Example: 0/0/4 my device name 4
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"

5.2.10. Parameter IA name 5

IA and name of the individual address, translating the IA to this name.
Example: 0/0/5 my device name 5
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"

5.2.11. Parameter IA name 6

IA and name of the individual address, translating the IA to this name.
Example: 0/0/6 my device name 6
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"

5.2.12. Parameter IA name 7

IA and name of the individual address, translating the IA to this name.
Example: 0/0/7 my device name 7
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"

5.2.13. Parameter IA name 8

IA and name of the individual address, translating the IA to this name.
Example: 0/0/8 my device name 8
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"

5.2.14. Parameter IA name 9

IA and name of the individual address, translating the IA to this name.
Example: 0/0/9 my device name 9
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"

5.2.15. Parameter IA name 10

IA and name of the individual address, translating the IA to this name.
Example: 0/0/10 my device name 10
used Pattern: "[0-9]+/[0-9]+/[0-9]+ [a-zA-Z 0-9]+"