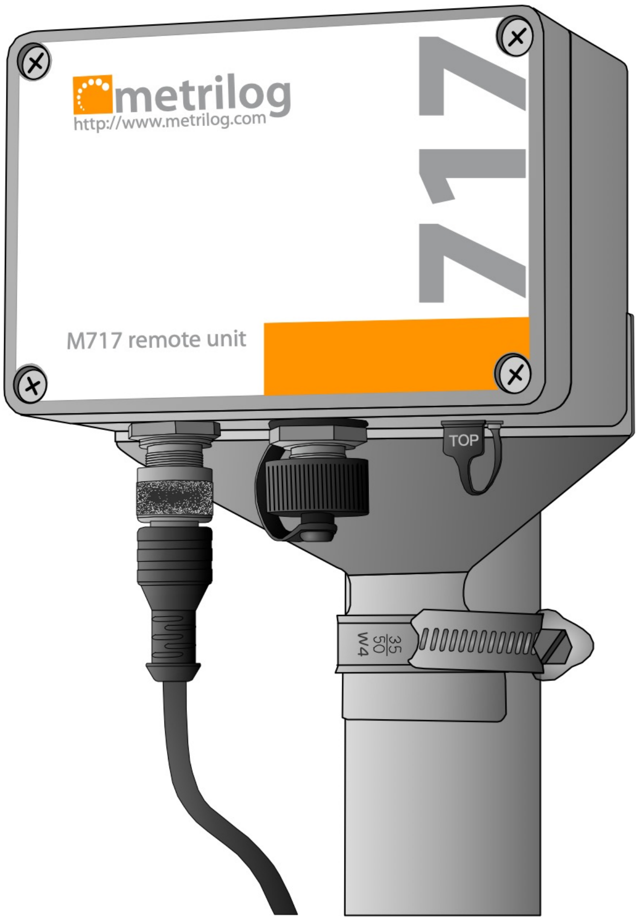


# Remote Terminal Unit M717

User Manual, firmware version 3

## Copyright Notice

All trademarks used throughout this document are the property of their respective owners. Mention of third-party products is for informational purpose only and constitutes neither an endorsement, nor a recommendation. Metrilog assumes no responsibility with regard to the performance or use of these products.

Document rev 3.0, issued 26 Aug 2025. This document is based on the M717 firmware version 3.0.0. In order to improve the performances of the product, Metrilog may issue from time to time firmware updates. The updates may add, modify or remove certain commands and functionalities, as described in this document. Metrilog assumes no obligation to update the manual as a consequence.

Metrilog Data Services GmbH, Office Park 4, C38, Vienna Airport City, 1300 Wien-Flughafen, Austria
www.metrilog.com

# Table Of Contents

# 1. Introduction

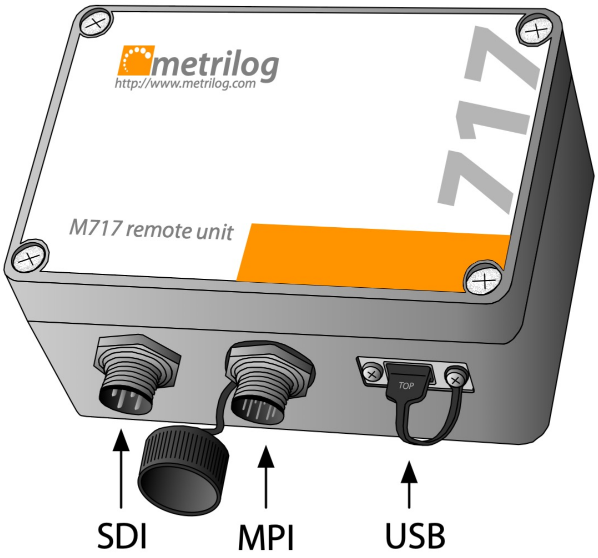
## 1.1. Product Overview

The Remote Terminal Unit (RTU) model M717 is a low power GSM/GPRS/UMTS/LTE based communication device, that includes flexible bus/serial interfaces and a data-logging unit. The M717 RTU supports following protocols and Input/Output (I/O) interfaces (see figure below):

- SDI, can be switched under software control between SDI-12 native, RS-485 or CAN;
- MPI, or Multi Protocol Interface; can be switched under software control between RS-232, RS-422 and RS-485;
- USB micro AB.

The M717 RTU can operate with multiple data sources, even simultaneous on both interfaces (e.g., several SDI-12 sensors on SDI and a Thies DL16 data logger on MPI). The M717 RTU supports a maximum of 50 sensors with a total of max. 128 sampled values (tags) per data record. The SDI-12 implementation also supports output tags by using specially designed, custom "X" commands (e.g., for switching valves). The Davis and Thies implementations support one sensor (the data logger) with a total of max. 128 sampled values (tags) per data record. The internal data logging memory (FIFO—First In, First Out) can store up to half a million individual values, the older values being overwritten when the memory fills up.

The SDI-12 implementation conforms to the SDI-12 specification version 1.3. For additional information on the SDI-12 bus, please consult the following document: "SDI-12, A Serial-Digital Interface Standard for Microprocessor-Based Sensors, Version 1.3". The document can be found on the SDI-12 Support Group's web site at http://www.sdi-12.org.



## 1.2. Safety

Please observe the following instructions:

- Do not open the unit, except if so instructed by the Metrilog technical personnel!
- Should it be necessary to open the unit (e.g., to replace the SIM card), the operation must be executed only in a moisture free environment. You will need a Philips screwdriver size PH2. Note that the screws remain attached to the lid, there is no need to remove them. Remove the lid slowly and carefully, because an antenna is fixed on its back. The antenna cable goes to a connector on the electronics board and it can be easily damaged. Before mounting the lid back, make sure the gasket is properly seated! If the gasket is damaged, contact Metrilog for a replacement. A defective gasket will lead to moisture entering the housing, thus permanently damage the unit.
- Please make sure that all the cables and connectors are properly fasten to avoid moisture entering the unit and/or the connectors.
- During normal use outdoors, the dust cover attached to the unit must always be inserted into the USB connector (see picture above).
- If either the MPI or the SDI connector is not used, it must be covered with the attached protection cap.
- The unit can be powered on all three connectors, even simultaneously, in which case the power source with the highest voltage will take precedence.
- Avoid attaching or detaching the RTU to a live (powered) bus or data logger, as this can lead to the damage of the RTU's data lines; the RTU is not hot-pluggable.
- The minimum voltage is 6 Volt for the SDI and MPI connectors and 4.7 Volt for the USB connector; the maximum voltage is 30 Volt for the SDI and MPI connectors and 5.3 Volt for the USB connector. Under no circumstances should the maximum voltage be exceeded!
- Improper commands issued through the USB port or over telnet, in particular commands modifying certain attributes, may compromise the functionality of the RTU and may even render it inoperable.

## 1.3. Information to Users in the US

- This device is granted permission to be operated in the US under the FCC ID **2A6UM-M717D**.
- Any changes or modifications not expressly approved by the party responsible for compliance could void the user's authority to operate the equipment.
- FCC RF Exposure statement: This device contains a RF Transmitter that complies with FCC RF radiation exposure limits set forth for an uncontrolled environment. This transmitter must be installed to provide a minimum separation distance of at least 20 cm from all persons and must not be co-located or operating in conjunction with any other antenna or transmitter, except in accordance with FCC multi-transmitter product procedures.
- This device complies with part 15 of the FCC Rules. Operation is subject to the following two conditions: (1) This device may not cause harmful interference, and (2) this device must accept any interference received, including interference that may cause undesired operation.

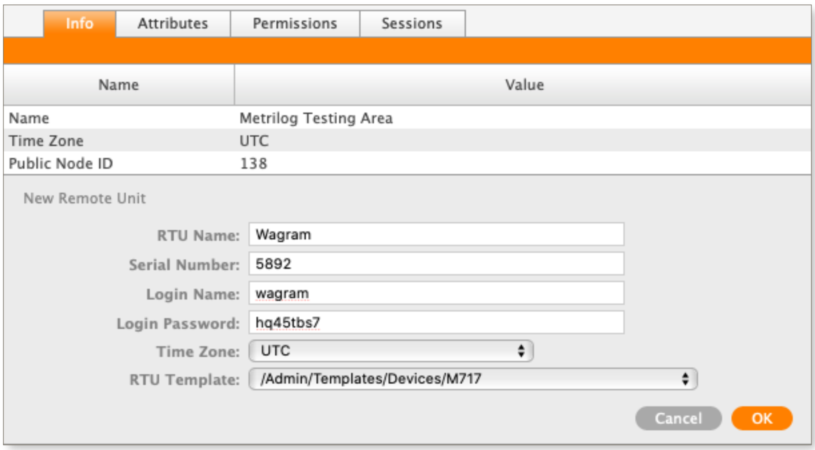
# 2. Installation

This section describes the installation of a new unit. It is recommended first to set-up the new RTU on the Metrilog M2M Gateway and then to proceed with the hardware installation.

## 2.1. Register the RTU to the M2M Gateway

The M717 RTU has been designed to operate in conjunction with the Metrilog M2M Gateway and Metrilog M2M Services. Before installing a M717 RTU in the field, it should be registered on the Metrilog M2M Gateway. Proceed as follows (you need administrator rights to add an RTU):

- Log-in on to the Metrilog M2M Gateway at https://www.metrilog.net (you need an account with administrative rights to do this);
- If you have sub-realms or sub-areas, navigate to the realm or area where you want to add the new RTU;
- Select the realm or area in the left pane of the Web interface and select "New Remote Unit" on the right pane;
- Fill in the relevant information; you will find the RTU's "Serial Number" on the metallic label affixed on one side of the M717 RTU;
- Choose a suitable name for the "RTU Name", usually a name representative for the location the RTU is installed at;
- The "Login Name" and "Login Password" are used later by the RTU to authenticate to the M2M Gateway; you can enter whatever name and password you deem appropriate, however it is recommended to use a name similar to the RTU name (max. 10 characters); the password should be at least 8 characters long and use both letters and numbers;
- Select the appropriate RTU template (e.g. M717);
- Click OK to finish.



| Name | Value |
|---|---|
| Name | Metrilog Testing Area |
| Time Zone | UTC |
| Public Node ID | 138 |

New Remote Unit

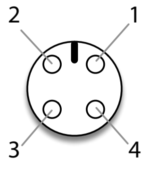
| | |
|---|---|
| RTU Name: | Wagram |
| Serial Number: | 5892 |
| Login Name: | wagram |
| Login Password: | hq45tbs7 |
| Time Zone: | UTC |
| RTU Template: | /Admin/Templates/Devices/M717 |

*Note: While you can use any character for the "RTU Name" field, do not use special characters as ü, ö, è, etc., for the "Login Name" and "Login Password" fields.*

Although it is not required for the proper operation of the RTU, you will need to configure sensors and tags to the RTU, depending on what sensors you plan to install in the field. A large variety of existing templates are available on the M2M Gateway, while new sensors/templates can be easily defined by the user.

## 2.2. The SDI Connector

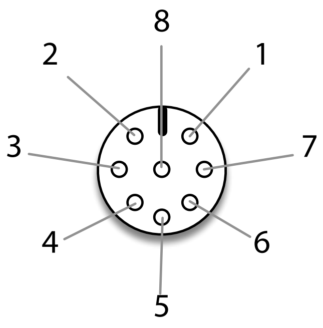
The pin-out of the SDI-12/RS-485 connector is shown below.

| | | | |
|---|---|---|---|
|  | 1 | Brown | + Vin (6 to 30 Volt) |
| | 2 | White | - Vin and SDI-12 GND |
| | 3 | Blue | RS-485 A/CAN-L (unused in native SDI-12 mode) |
| | 4 | Black | SDI-12/RS-485 B/CAN-H |

*Note: The colours given in the last column are valid for the standard Metrilog M12 cable (option). Pin 3 is not used in native SDI-12 mode.*

## 2.3. The MPI Connector

The pin-out of the Multi Protocol Interface (MPI) is given below.

| | | | |
|---|---|---|---|
|  | 1 | White | - Vin and GND |
| | 2 | Brown | + Vin (6 to 30 Volt) |
| | 3 | Green | RS-232 RxD/RS-485 A |
| | 4 | Yellow | RS-232 TxD/RS-422 Z |
| | 5 | Grey | RS-485 B |
| | 6 | Pink | RS-232 GND |
| | 7 | Blue | RS-422 Y |
| | 8 | Red | GND/Shield |

*Note: The colours given in the last column are valid for the standard Metrilog M12 cable (option).*

## 2.4. The Micro USB Service Connector

The USB connector is used for maintenance and service. You should not have to deal with this connector unless indicated so by the Metrilog support personnel. For additional information on the use of this connector, check also the "Commands" section of this manual.

## 2.5. About the Data Acquisition Subsystem

Depending on the interface to the input/output devices (sensors), the M717 RTU supports the following sensors and data loggers:

- Sensors based on the SDI-12 protocol, version 1.3. It has various operating modes and can be configured to accept a large variety of sensors both in native SDI-12 hardware interface and the RS-485 balanced interface. The RS-485 interface allows for longer cables, but unfortunately not many sensor manufacturers implement SDI-12 over RS-485.
- Davis Instruments, Inc. Vantage Pro weather stations with the WeatherLink interface (also from Davis Instruments). The communication is done through an RS-232 link over the MPI connector.
- A series of data loggers manufactured by Adolf Thies GmbH & Co. KG. (e.g., models TDL14 and TDL16). The RS-232, RS-422 and RS-485 interfaces over the MPI connector are supported.

Specific functions of the Data Acquisition subsystem are configured by means of attributes. The subsystem is hierarchically structured, and it can have a number of sensors, each of them in turn having a number of tags. A tag in this sense represents a specific value that can be read from — e.g., a temperature, or written to — e.g., a coil or a valve. In other words, there are input and output tags.

*Note: Currently, only the SDI-12 protocol supports output tags (by means of SDI-12 "X" commands); the Davis and Thies data loggers support only input tags.*

Although there are some common attributes, depending on the attached sensors and/or data logger, specific attributes are used to define the functionality of the sensors and tags. For instance, *acquisitionMode* and *acquisitionSchedule* are common attributes for all kind of sensors; however, *sdiMethod* is specific to the SDI-12 sensors, while *archiveInterval* is specific to the Davis data logger.

The Data Acquisition Subsystem includes a First-In, First-Out (FIFO) ring storage. The data provided by the sensors is stored in the FIFO until delivered by the communication subsystem via the Internet. The data is stored persistently and is not lost while the RTU is powered down.

## 2.6. SDI-12 Sensors

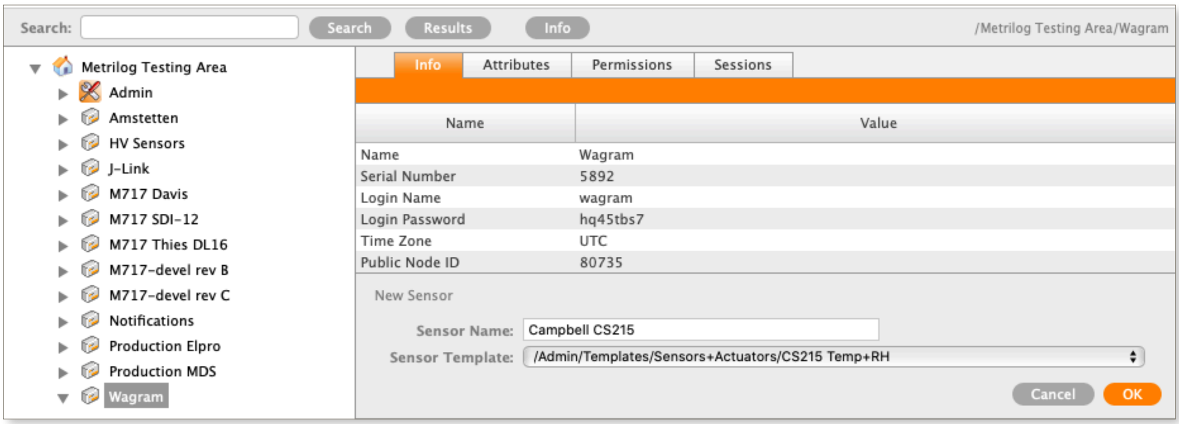
The M717 RTU uses a four-wire cable carrying the SDI-12 bus signals and the bus power supply (typically 12 Volt). Alternatively the bus can be switched to RS-485 levels, but this is allowed only if all devices on the bus support it. You cannot mix SDI-12 and RS-485 sensors on the same bus.

The switch from SDI-12 (default) to RS-485 is done automatically depending on the wiring of the sensor: if the blue wire (pin 3) is used, then the bus is operating in RS-485 mode. For SDI-12 native sensors, only the pins 1, 2 and 4 of the connector must be used (the brown, white and black wires of the cable), while pin 3 (blue wire) must be left open.

*Note: CAN mode is not supported by the current firmware, but may be in a future update.*

Depending on the sensor, a template may already be available on the M2M Gateway. To add a new sensor to an RTU follow the steps below:

- Log in to the M2M Gateway and select in the left pane the RTU you want to add a new sensor to.



| Name | Value |
|---|---|
| Name | Wagram |
| Serial Number | 5892 |
| Login Name | wagram |
| Login Password | hq45tbs7 |
| Time Zone | UTC |
| Public Node ID | 80735 |

New Sensor

Sensor Name: Campbell CS215

Sensor Template: /Admin/Templates/Sensors+Actuators/CS215 Temp+RH

- In the right pane, click the “New Sensor” button.
- Name your sensor, then from the drop down select the template corresponding to the sensor you want to add.
- Click the "OK" button. The sensor will be added to the RTU.

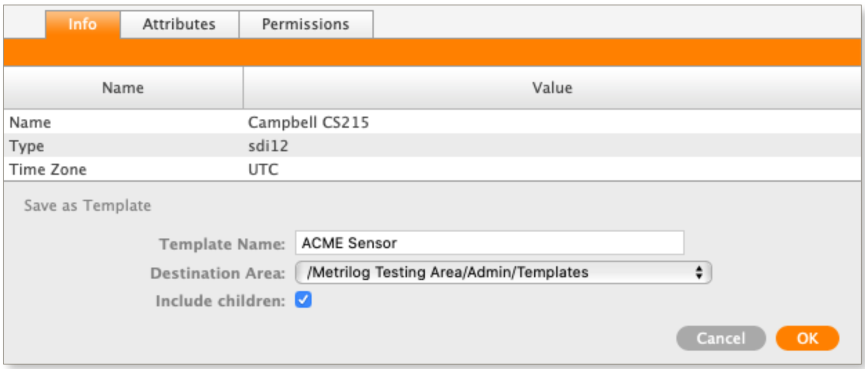
After the sensor has been added, select it in the left pane and click “Attributes” to complete or change some attributes. Verify if the *sdiAddress* is empty or incorrect, it must be identical to the SDI-12 address of your sensor. Verify also that the *sdiMethod* suits the one you want to use with your sensor, as it can be either “M”, “MC”, “C, “CC”, or a “Rx” variant (for more details, consult the SDI-12 specification, as well as the sensor’s User Manual).

The final step is to verify/update the tags’ attributes, in this case only the *sdiIndex*. If the attribute had a value given during the template generation, then it should be OK. However, if it is missing, or your SDI-12 sensor has a different configuration, you must set it here. Essentially, the *sdiIndex* is the position of the sensor’s value in the SDI-12 “Dx” response string. Note that the index numbering starts with 0.

If there is no template for your sensor already, you must create one by yourself. There are two methods: either you use an existing template and modify it to suit the new sensor (recommended, as it is easier), or you create one from scratch. Both methods are described in the following sections.

## 2.6.1. Create a New Template From an Existing One

This method is the easier one. Do the following:



| Name | Value |
|---|---|
| Name | Campbell CS215 |
| Type | sdi12 |
| Time Zone | UTC |

Save as Template

Template Name: ACME Sensor

Destination Area: /Metrilog Testing Area/Admin/Templates

Include children: ☑

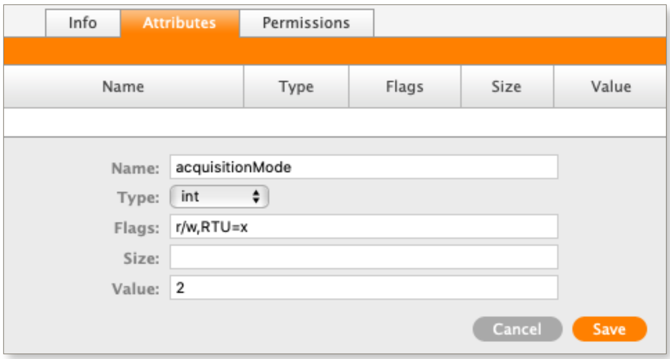
- Log in to the M2M Gateway and in the left pane click and expand the “Admin” entry, then click the “Templates” entry.

- Select an existing sensor template which is as much as possible close to the one you want to create.
- In the right pane, click the “Save as Template” button.
- In the following dialog box, enter the name of the new template and select the destination area (where the template will be saved on the server).
- Check the “Include children” checkbox if you want to take over the tags of the original template (recommended).
- Click "OK".
- Select the new template and verify/modify its attributes according to the parameters of the sensor.
- Examine the children tags and delete the ones you don’t need, or add new tags according to the hardware configuration of your sensor.
- Verify and modify accordingly the attributes of all the tags of the new sensor.

In a similar fashion you can create new tag templates, based on existing templates (use the “Save as Template” button).

### 2.6.2. Create a New Template From Scratch

This method requires more work. Proceed as follows:



Name: acquisitionMode
Type: int
Flags: r/w,RTU=x
Size:
Value: 2

- Log in to the M2M Gateway and in the left pane click and expand the “Admin” entry, then click the “Templates” entry.
- On the right pane, click the “New Template” button.
- Name the new template; use a name that can easily identify the new sensor, usually the manufacturer and the type of the sensor.
- Select from the Template Class “sensor” and click "OK". The new template will be generated, however it will be empty.
- Select the newly generated template and click the “Attributes” tab. You will now add several attributes.
- Click the “New” button; in the dialog box that appears you must define the attribute. For a proper operation of the RTU, a list with the minimum of attributes required and their parameters follows:
    - *acquisitionMode*, type: *int*, flags: *r/w,RTU=x*, Size: , Value: *2*
    - *acquisitionSchedule*, type: *string*, flags: *r/w,RTU=x*, Size: *100*, Value: **/10*
    - *iconName*, type: *string*, flags: *a/a*, Size: *20*, Value: *COMBO*
    - *sdiAddress*, type *string*, flags: *a/a,RTU=x*, Size: *1*, Value:
    - *sdiInfo*, type: *string*, flags: *r/-*, Size: *40*, Value:
    - *sdiMethod*, type:*string*, flags: *a/a,RTU=x*, Size: *3*, Value:

*Note: The attribute names, as well as all the parameters must be entered exactly as written; where no value is given, the field should be left empty. However, you may enter default values for the sdiAddress and sdiMethod (consult the sensor’s specifications).*

After all the sensor’s attributes have been entered, you need to define and add the sensor’s tags.

- Select the sensor and click the “New Tag” button.
- Enter the name of the new sensor (e.g., “Temperature”), then select the appropriate sensor template from the drop-down.
- Finally click the "OK" button.

As with the sensors, if you can’t find a template already defined for your tag, you must define one by yourself. This is similar to the creation of a sensor template, except that the attributes are different. Proceed as follows:

- Log in to the M2M Gateway and in the left pane click and expand the “Admin” entry, then click the “Templates” entry.
- On the right pane, click the “New Template” button.
- Name the new template; use a name that can easily identify the new tag, e.g., “Temperature”.
- Select from the Template Class “tag” and click "OK". The new template will be generated, however it will be empty.
- Select the newly generated template and click the “Attributes” tab. You will now add several attributes.
- Click the “New” button; in the dialog box that appears you must define the attribute. For a proper operation of the RTU, a list with the minimum of attributes required and their parameters follows:
    - *addUPItemplate*, type: *string*, R/-, Size: *30*, Value: *addUPI_ANALOG_TAG_V1_0*
    - *EUID*, type: *int*, flags: r/-, Size: , Value: *<see note below>*
    - *iconName,* type: *string*, flags: *a/a*, Size: *20*, Value: *<see note below>*
    - *maxValue,* type: *double*, flags: *r/-*, Size: , Value: *<see note below>*
    - *minValue,* type: *double*, flags: *r/-*, Size: , Value: *<see note below>*
    - *sdiIndex,* type: *int*, flags: *a/a,RTU=x*, Size: , Value:

*Note: The addUPItemplate is not required by the RTU, but it is used by certain application software as e.g. “Instruments”. It must be entered exactly as shown. The EUID (Engineering Unit ID) is a conventional rule that associates measurement units (e.g., meter, second, gram, etc.) to an ID (a number). The RTU doesn’t need it, but it is helpful for application software, as well as for the M2M Gateway, to associate the proper measurement unit to a tag. The iconName is not required by the RTU; it is a reference to an icon that will be displayed by the M2M Gateway user interface. Both the proper EUID and the iconName can be found clicking on the “Info” button on the main menu bar (see picture below). The maxValue and minValue are not required by the RTU, but are used by the application software, for instance when setting the limits of a graph. These values must reflect the maximum and minimum values that can be delivered by the sensor.*

*Note: The sdiIndex is the position of the sensor’s value in the SDI-12 “Dx” response string; the index numbering starts with 0.*

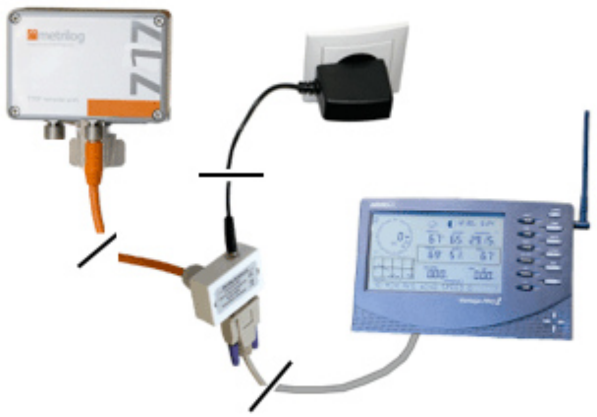
## 2.7. Interface to a Davis Vantage Pro Console

The following instructions are valid when the M717 RTU is connected to a Vantage Pro weather station (manufactured by Davis Instruments). The M717 RTU can be installed both indoors and outdoors.

*Note: The M905 Connexion Box, the Davis Console, as well as the mains adapter are not intended for outdoors installation!*

To connect the M717 RTU to the Vantage Pro Console, a WeatherLink data cable from Davis Instruments is required. Note that for this configuration the data collected by the Davis Console is already logged into the WeatherLink data logger, thus the data passes through two levels of storage: one offered by the Davis System and a second by the M717 RTU.

The following diagram depicts the connections between the M717 RTU, the M905 connexion box and the mains adapter.



The configuration of the Davis Console on the M2M Gateway is done as for the SDI-12 sensors, but it is considerably simplified as a suitable template is already available.

## 2.8. Interface to a Thies TDL14 or DL16 Data Logger

The Thies data loggers implement a proprietary serial protocol. The TDL14 logger supports the RS-232 and RS-485 full duplex modes, while the DL16 logger additionally supports the RS-485 half-duplex mode. The communication with the logger is done using a 10 m cable attached to the MPI connector.

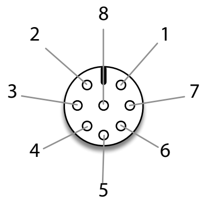
Before proceeding with the actual installation in the field, it is recommended to verify, and possibly modify the specific attributes that define the MPI functionality. Proceed as follows (you need administrator rights to add an RTU):

- Log-in on to the Metrilog M2M Gateway at https://www.metrilog.net.
- Navigate to the RTU that must be verified/modified and select it.
- On the right pane, select the tab "Attributes".
- The attribute *mpiBaudRate* determines the baud rate of the MPI interface: set it to fit the baud rate of the data logger (default 9600 Baud; the DL16 supports also higher baud rates).

- The attribute *mpiMode* determines the operation mode of the MPI interface: set to one of RS-232, RS-422 or RS-485, depending on the configuration of the data logger.
- The attribute *mpiTermination* is effective only in the RS-422 and RS-485 modes: if required, 120Ω termination resistors can be activated on the M717 RTU side of the bus.
- After all attributes are properly set, click the "Save" button.

To connect the 10 m M12 cable, you must open the Thies data logger and insert it through a free cable gland. Attach the cable wires to the corresponding screw posts as shown in to the table below, depending on the selected serial mode (RS-232, RS-422 or RS-485) and the logger type.

*Note: If you are performing the operations while the Thies data-logger is powered, make sure the other end of the cable is* ***not*** *connected to the M717!*

| M717 Pin-out | | | TDL14 | | DL16 | | |
|---|---|---|---|---|---|---|---|
| | | | RS-232 | RS-422 | RS-232 | RS-422 | RS-485 |
|  | 1 | White | 6 (GND) | 6 (GND) | 1 (GND) | 1 (GND) | 1 (GND) |
| | 2 | Brown | 5 (+12V) | 5 (+12V) | 7 (+12V) | 7 (+12V) | 7 (+12V) |
| | 3 | Green | 11 (TxD) | 12 (TxD+) | 6 (TxD) | 5 (TxD+) | N.C. |
| | 4 | Yellow | 9 (RxD) | 10 (RxD-) | 2 (RxD) | 3 (RxD-) | 6 (B) |
| | 5 | Grey | 11 (TxD) | 14 (TxD-) | N.C. | 6 (TxD-) | N.C. |
| | 6 | Pink | 13 (GND) | 13 (GND) | 4 (GND) | 4 (GND) | 4 (GND) |
| | 7 | Blue | N.C. | 8 (RxD+) | N.C. | 2 (RxD+) | 5 (A) |
| | 8 | Red | N.C. | N.C. | N.C. | N.C. | N.C. |
| | Shield | | Connect to metal housing | | | | |

*Note: The wires marked in the table as "N.C." must be left unconnected and insulated from other wires and/or metallic parts.*

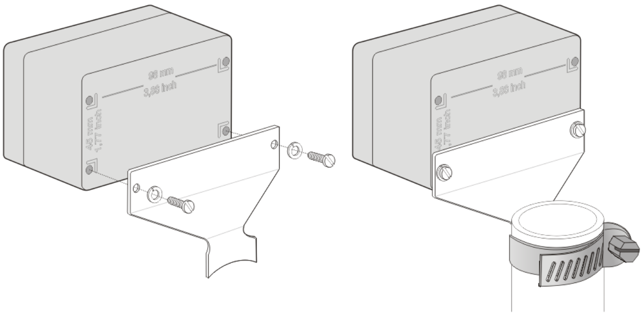
After all the wires have been properly connected, you may now attach the other end of the cable to the M717 unit. Please observe the following:

- Before connecting the M717 cable wires to the screw posts, make sure there are no other wires already connected! The M717 cannot be operated in parallel to another device/modem.
- The cable shielding must be connected to the metallic housing of the logger. If the cable gland does not provide a ground connection to the cable's shield, a separate wire should be used to connect it to the ground.

The configuration of the Thies Data Logger on the M2M Gateway is done similarly to the SDI-12 sensors, but it is considerably simplified as suitable templates for standard configurations of TDL14 and DL16 are already available. Additional tags may be defined and added according to the actual logger configuration.

## 2.9. Mechanical Installation

The M717 unit should be preferably mounted outdoors on a 40 mm diameter mast or support by means of an (optional) metallic mounting fixture and a hose clamp. In this case fasten first mounting fixture to the M717 enclosure using two M6 screws and spacers (see the figure below). If the unit is not installed outdoors, rather placed on a table or in a cabinet, the metallic fixture may not be required.



It is recommended to place the top of the unit (where the internal antennas reside), not too close to metallic objects, especially to the mast itself. Ideally the unit should be mounted at the top of the mast. Use tie wraps to fasten the cables to the mast after the mechanical installation is completed.

If the unit is kept indoors, try to place it close to a window if the cellular signal is weak.

## 2.10. Operation

After being powered up, an uninitialised RTU will first check the SIM card and extract the provider specific information (APN, Name, Password) from it. This information, together with the initial address of the Metrilog M2M Gateway is stored in a table in the RTU's firmware.

Note: *The RTU must be already registered on the Metrilog M2M Gateway as described at the beginning of this section. In addition, the SIM card must be active. If not sure, please contact Metrilog.*

At this point, the RTU is able to connect via the cellular network to the M2M Gateway. If the connection succeeds, the RTU retrieves its configuration, including its account data and the attached sensors. The account data is based on the credentials offered by information contained in the SIM card, the IMEI, and/or the serial number of the device. If the SIM card is not known to the M2M Gateway, the RTU will not be able to log in and retrieve configuration and account data. The error messages in the log file document such situations, as well as other similar potential issues. You can see the log files, if you have direct access to the unit, by connecting it to a PC using the USB connector. For more details see the "Commands" section.

# 3. Configuration

The M717 RTU is automatically configured by the Metrilog M2M Gateway, therefore in most cases no user intervention is necessary. However, the operation of the RTU and the sensors can be customised to a certain extent through the M2M Gateway.

## 3.1. The Metrilog M2M Gateway

An important service offered by the M2M Gateway is the handling of the configuration of the RTUs. The complete behaviour of an RTU is defined on the Gateway. The RTU parameters can be defined and/or modified at any time on the M2M Gateway after which they are retrieved by the RTU as soon as a connection is made.

A connection can only be initiated by the RTU and not the other way round. That means "polling" the RTU is not possible. In order to overcome this issue, the M717 RTU implements a flexible scheduler that specifies when and how often the RTU should initiate a connection to the M2M Gateway. As with many other parameters on the RTU, the scheduler is remotely configurable through the M2M Gateway.

By changing a parameter on the Gateway, a "task" will be generated on the server and added to a queue for that particular RTU. When the RTU connects, the task will be passed along and the RTU will execute and acknowledge it.

The functionality of an RTU is defined by a collection of attributes. A list with a partial description of those attributes is given in the "Attributes" section of this document. The Web based User Interface of the M2M Gateway offers an easy and intuitive method to add/modify/delete attributes. However, standard devices (e.g., the M717 RTU family, as well as a large collection of sensors), are already predefined through templates, so you should not need to add or change attributes.

*Note: Improper manipulation of the attributes can lead to malfunction of the RTUs. You must first understand what function a certain attribute performs before attempting to change its value.*

## 3.2. The USB Service Port

The use of the service port is recommended only in special cases during an installation operation where there is no Internet access, or for some reasons the RTU can't contact the M2M Gateway.

To communicate with the RTU through the service port, an USB-A to USB micro-AB cable and a PC with a terminal program is required (e.g., *Hyperterminal*, *minicom*, *tio*, etc.). Depending on the operating system running on your PC, you might need to install an USB-CDC driver, but on most modern operating systems this driver is already installed.

Connect a micro-USB to A cable to a PC and use the new virtual serial port (COM or tty) registered in your system with the terminal program. You should see in your terminal program the RTU name followed by a login prompt: enter the RTU password (you can find it by logging into the M2M Gateway, navigating to the respective RTU and looking it up under the tab *Attributes* as *cliPasswd*) then the ":" prompt should appear.

## 3.3. Telnet

The M717 RTU has also a built-in telnet server. However, to access the CLI over telnet, there are several prerequisites:

- The telnet server is by default not active, therefore it must be activated through the *cliOverIP* attribute. This is done through the M2M Gateway Interface. At the next session, the telnet server will be started and will remain active for 5 minutes (default value).
- The telnet server will not be activated if the RTU can't connect to the Internet.
- Currently only units using the Global SIM cards provided by Telekom Austria can be reached via telnet.
- For security reasons, all the units using the Telekom Austria SIM cards are part of a private network that can't be accessed directly from the Internet. To reach these units you need a special cellular router using also a SIM card from Telekom Austria that belong to the same VPN (please contact Metrilog in case you need one).
- The modem must be properly configured, integrated in your network, or at least connected over IP to your computer.

To reach the unit over telnet, after the telnet server on the RTU has been started, use a telnet client to connect to the IP address indicated by the attribute *assignedIP,* as shown on the M2M Gateway.

## 3.4. What if the CLI Password is Lost

To access the CLI, either over USB or telnet, you need the RTU password, which is registered with the RTU on the M2M Gateway. But what if the RTU has been deleted from, or replaced on the M2M Gateway and the password was not kept somewhere? You cannot access the RTU over USB or telnet anymore, not even for cleaning it up (i.e. to perform a factory reset). In this case you have two options:

- Contact Metrilog, and you will be instructed what to do.
- Register the RTU to the M2M Gateway using a temporary name (see the Register the RTU to the M2M Gateway section), then perform a factory reset by entering at the password prompt

  ```
  factory-reset
  ```

  instead of password.

By using the second option, all the data in the RTU will be deleted (including the database and the log entries). Depending on the selected configuration when registering the RTU under the temporary name, after 10 to 20 minutes you will be able to recover the RTU's password by looking it up in the *Attributes* tab, under *cliPasswd.*

# 4. Commands

The information in this chapter is intended for well trained technicians to pinpoint errors or device malfunctions.

To access the CLI, make sure to have the RTUs password at hand, it can be found on the M2M Gateway by selecting the respective RTU, then clicking on *Attributes* and looking up the *cliPasswd* attribute.

Once the CLI is reached using one of the methods described earlier (USB or telnet), and after entering the correct password, a ":" (colon) prompt will be displayed on the terminal. The commands can be typed in lower or upper case.

*Note: An exception to this is when typing attribute names and values, they are case sensitive.*

## 4.1. General Commands

### 4.1.1. help

**Description**

Lists all the available commands with a short description. It is a useful command if you don't remember the syntax of a command; for further information on a specific command, type <*command*> *-h*, this will show the subcommands and/or the parameters accepted by a command.

### 4.1.2. ver

**Description**

Returns the versions of the application and boot-loader of the unit, as well as general information about the hardware.

**Example**

```
: ver
Model M717, rev. D
M717-app, ver. 3.0.0 (build 1917), Aug 08 2025, 19:09
M717-boot, ver. 1.2.0 (build 279), Aug 06 2025, 15:58
Core clock 96 MHz, core temperature 30 deg.C, Vdd 3.01 Volt
External power supply 12.4 Volt, backup battery 3.18 Volt
(c) 2018-2025 Metrilog Data Services GmbH
```

### 4.1.3. echo

**Description**

Enables/disables echoing the typed characters. Issued without a parameter, it returns the current state. Default for *echo* is ON.

**Remarks**

In telnet mode, the echo is switched off, as the telnet client will normally echo the characters.

**Examples**

```
: echo
Echo is on

: echo off
```

### 4.1.4. ps

**Description**

Shows the RTOS threads, their state and priority, CPU load, stack use and memory available. This command is informative.

**Usage**

ps

**Example**

```
: ps
Thread Name             State  Prio   %CPU   Stack
===========             =====  ====   ====   =====
db-buffer               Wait    96    < 1%    1092
cli-manager             Wait    96    < 1%     928
cli-cdc1                Run     96      1%    2532
comm-worker             Wait    96    < 1%     276
mpi-worker              Wait    96    < 1%    2820
sdi-worker              Wait    96    < 1%    1204
location-worker         Wait    96    < 1%    1168
cron                    Wait    96    < 1%    1124
log-buffer              Wait    96    < 1%    1188
main                    Wait    32    < 1%    1760
tcpip                   Wait    64    < 1%    3156
idle                    Ready   16     98%     892
Heap: 135KB free out of 209KB available
Uptime: 12d, 3h, 45m
```

### 4.1.5. passwd

**Description**

Change the CLI password.

**Usage**

passwd [-h ]
The *-h* option shows a short help text.

**Remarks**

Note that the passwords are auto-generated and then uploaded to the M2M Gateway using a secure channel (HTTPS). The passwords cannot be user defined.

**Examples**

```
: passwd -h
Usage:    passwd

: passwd
Are you sure you want to generate a new password? (y/n): y
New password: 7CbJjB9slgw
```

### 4.1.6. log

**Description**

Shows log entries and log statistics.

**Usage**

log [ dd/mm/yy [ hh:mm:ss ]] to show the log entries
log stat to show the log system information
log clear [ -y ] to clear the log

**Remarks**

The date/time parameter must be specified only once, the system will display the next 20 lines after the date given (if no, or less than 20 entries are available, there will be no, or only the available entries displayed). To continue with the next log entries, enter only the *log* command, without parameters, until all entries are displayed.
The *-y* option is used to override the confirmation question, the log will be immediately cleared.

**Examples**

```
: log 02/08/2020
02/08/2020 09:59:09 [     4] cli_over_ip_manager(): console started on /dev/ptty0
02/08/2020 10:00:10 [     4] cli_over_ip_manager(): console on /dev/ptty0 terminated by the user
02/08/2020 10:12:26 [     1] reset_handler(): system halted
02/08/2020 10:12:29 [     1] reset_handler(): bootloader started after software reset
02/08/2020 10:12:29 [    64] init_block_devices(): file system successfully mounted
02/08/2020 10:12:29 [  2048] update(): update pack found at /flash/firmware/m717_1_04.pack
02/08/2020 10:12:31 [  2048] update(): M717-app update succeeded
02/08/2020 10:12:32 [     1] reset_handler(): system halted
02/08/2020 10:12:34 [     1] reset_handler(): system started after software reset
02/08/2020 10:12:34 [     1] init_block_devices(): database opened in 94.39 ms
02/08/2020 10:12:34 [    64] init_block_devices(): file system successfully mounted
02/08/2020 10:12:34 [  1024] init_config(): invalid configuration, attempting flash restore
02/08/2020 10:12:34 [    64] rpc_config_struct_load(): configuration loaded
02/08/2020 10:12:34 [  1024] init_config(): flash restore succeeded, configuration OK
02/08/2020 10:22:29 [     1] reset_handler(): system halted
02/08/2020 10:22:32 [     1] reset_handler(): bootloader started after software reset
02/08/2020 10:22:32 [    64] init_block_devices(): file system successfully mounted
02/08/2020 10:22:32 [  2048] update(): update pack found at /flash/firmware/m717_1_04.pack
02/08/2020 10:22:35 [  2048] update(): M717-app update succeeded
02/08/2020 10:22:35 [     1] reset_handler(): system halted
: log
02/08/2020 10:22:37 [     1] reset_handler(): system started after software reset
02/08/2020 10:22:37 [     1] init_block_devices(): database opened in 94.40 ms
02/08/2020 10:22:37 [    64] init_block_devices(): file system successfully mounted
02/08/2020 10:22:37 [  1024] init_config(): configuration OK
```

```
: log stat
The write pointer is at block 8
Oldest: Wed Jul 22 23:25:55 2020
Newest: Tue Aug  4 10:49:05 2020
```

### 4.1.7. attr

**Description** Used to show attributes and their values.

**Usage** attr [ -a | -s | -t | -h ] to show attributes, *-a* for all, *-s* for all sensors, *-t* for all tags
attr [ nodeid ] attrib_name to show a specific attribute; if *nodeid* is not specified, the RTU node is assumed.

**Remarks** The *-h* option shows a short help text.

**Examples**

```
: attr gsmAllowRoaming
true

: attr -s

    *** sensor ID (index): 79630 (0)
                     type: Intern
          acquisitionMode: 2
      acquisitionSchedule: 0
                     info: M717 rev. C Internal Sensor
                 lastDate: Mon Jun 29 15:00:00 2020
              putdataDate: Mon Jun 29 15:00:00 2020

   *** sensor ID (index): 77178 (1)
                     type: SDI-12
          acquisitionMode: 2
      acquisitionSchedule: */10
               sdiAddress: 0
                sdiMethod: CC
```

```
            sdiPostmethod:
                 sdiDebug: false
                  sdiInfo: 13 CSL    CS215 2.0
                 lastDate: Mon Jun 29 15:30:00 2020
              putdataDate: Thu Jan  1 01:00:00 1970

: attr 77178 sdiMethod
CC
```

### 4.1.8. hwid

**Description** Shows the hardware ID of the RTU.

**Usage** hwID

**Remarks** The ID can be also shown using the *attr* command (*attr hwID*).

**Example**

```
: hwid
hwID is 5010
```

### 4.1.9. reboot

**Description** This command reboots the RTU.

**Usage** reboot [ -c | -cc | -y | -h ]
Without parameters, the system will reboot. With the *-c* parameter, the system reboots, but immediately after start-up the internal configuration will be erased. With the *-cc* parameter, the system reboots, but immediately after start-up both the internal configuration, as well as all the historians will be erased.
WARNING: after using the option *-cc*, all data in the RTU will be lost!
By using the option *-y*, the RTU reboots immediately, without asking for confirmation, while the *-h* option shows a short help text.

**Remark** This operation must be done whenever an RTU is relocated to another site with a different sensor configuration.

**Examples**

```
: reboot
Are you sure? (y/n) y
Please don't disconnect the power while the modem is shut down...
Done.
System will now restart

'Wagram' password:

: reboot -c
Are you sure? The configuration will be cleared (y/n): y
Please don't disconnect the power while the modem is shut down...
Done.
System will now restart, the configuration has been cleared

Type "help" for the list of available commands
:
```

### 4.1.10. exit

**Description** Exits the CLI. This command is particularly important when connected over telnet, to close the connection. It has no special meaning when connected over USB or the MPI connector. However, in all cases the connection to the CLI will be lost, and for USB specifically, the USB cable must be unplugged and plugged again to reconnect.

**Example**

```
: exit
Exiting…
```

## 4.2. Data Acquisition Commands

### 4.2.1. dacq

**Description** Commands operating on the Data Acquisition subsystem.

**Usage** dacq [ -h | subcommand ] with the *-h* parameter (or without parameters), a list of all data acquisition commands will be shown. The subcommands are described in the following sections.

### 4.2.2. dacq info

**Description** Get the sensor/logger information or version.

**Usage** dacq info <*sensor_IX*> where <*sensor_IX*> is the ID of the sensor to inquire.

**Remarks** The ID of a specific sensor can be found using the *attr -s* command, under the section "*sensor ID (index)*", it is the value given in parenthesis. If there is only one sensor configured in the RTU, then the ID is always 0.

**Examples**

```
: dacq info 0
13MetrilogM512rD1.5.7439#000000

: dacq info 1
13 CSL    CS215 2.0
```

### 4.2.3. dacq sample

**Description** Used to sample and show sensor/logger data.

**Usage** dacq sample <*sensor_IX*> [ date [ time ]] where <*sensor_IX*> is the ID of the sensor to sample, while the optional date and time parameters can be used to specify a point in time after which the data should be returned. Obviously, this is valid only for loggers, i.e., devices having storage possibilities (e.g., Thies loggers or Vantage Pro consoles); for SDI-12 sensors, these parameters are ignored.

**Remarks** The ID of a specific sensor can be found using the *attr -s* command, under the section "*sensor ID (index)*", it is the value given in parenthesis. If there is only one sensor configured in the RTU, then the ID is always 0.
The values in parenthesis represent the status bits: if 0, all is OK while status 1 means missing data.
Certain sensors may need longer time to return data, just be patient.

**Example**

```
: dacq sample 1
:
```

```
Mon Aug  3 18:27:45 2020
26.025999[0] 60.115002[0]
```

### 4.2.4. dacq retrieve

**Description**

Used to sample and store sensor/logger data.

**Usage**

dacq retrieve <*sensor_IX*> [ date [ time ]] where <*sensor_IX*> is the ID of the sensor to sample, while the date and time parameters can be used to specify a point in time after which the data should be returned, or in case of simple sensors (e.g. SDI-12), the date/time stamp which will be associated to the retrieved values when they are stored in the RTU's memory. If no date is provided, the current date and time will be used; for data loggers, the date of the last datapoint in the local memory will be used for the request and the returned data point's date will be used as timestamp.

**Remarks**

The ID of a specific sensor can be found using the *attr -s* command, under the section "*sensor ID (index)*", it is the value given in parenthesis. If there is only one sensor configured in the RTU, then the ID is always 0.
Note that this command, in contrast to the *dacq sample* command, stores the retrieved data to the internal data base of the RTU. This data will then be pushed on the M2M Gateway during the next session.
Certain sensors may need longer time to return data, just be patient.

**Example**

```
: dacq retrieve 0
```

### 4.2.5. dacq abort

**Description**

Aborts a running *dacq sample* or *dacq retrieve* transaction.

**Usage**

dacq abort <*sensor_IX*> where <*sensor_IX*> is the ID of the sensor to abort a running transaction.

**Remarks**

Not all sensors support an *abort* operation.
The ID of a specific sensor can be found using the *attr -s* command, under the section "*sensor ID (index)*", it is the value given in parenthesis. If there is only one sensor configured in the RTU, then the ID is always 0.

**Example**

```
: dacq abort 0
```

### 4.2.6. dacq date

**Description**

Use to set/synchronise the date/time of a sensor/logger.

**Usage**

dacq date <*sensor_IX*> where <*sensor_IX*> is the ID of the sensor to synchronise.

**Remarks**

Not all sensors support a *date* command, in particular it is not supported in a standard way by the SDI-12 sensors. Thies and Davis loggers support this command.
The ID of a specific sensor can be found using the *attr -s* command, under the section "*sensor ID (index)*", it is the value given in parenthesis. If there is only one sensor configured in the RTU, then the ID is always 0.

**Example**

```
: dacq date 0
```

### 4.2.7. dacq interval

**Description** Used to set the sampling interval of a sensor/logger.

**Usage** dacq interval <*sensor_IX*> <*interval*> where <*sensor_IX*> is the ID of the sensor to set the sampling interval to and <*interval*> is the new sampling interval, in seconds.

**Remarks** Not all sensors support an *interval* command, in particular it is not supported in a standard way by the SDI-12 sensors. Thies and Davis loggers support this command. The command determines the sampling and storage interval in the logger themselves.
Do not confuse this setting with the interval defined by a sensor's *acquisitionSchedule* attribute; for SDI-12 sensors, this attribute defines the storage interval in the RTU's memory.
The ID of a specific sensor can be found using the *attr -s* command, under the section "*sensor ID (index)*", it is the value given in parenthesis. If there is only one sensor configured in the RTU, then the ID is always 0.

**Example**

```
: dacq interval 0 600
```

### 4.2.8. dacq direct

**Description** Used to directly connect to a sensor.

**Usage** dacq direct <*sensor_IX*> where <*sensor_IX*> is the ID of the sensor to connect to. This command is interactive, to exit the command type <*ctrl-X*>.

**Remarks** While this command allows connection to any sensor, not all sensors support an interactive "conversation". In particular SDI-12 sensors cannot be reached in this way. Moreover, the sensor/logger commands must be known. Thies loggers and Davis consoles have their own proprietary protocols, for more details consult their respective user manuals.
The ID of a specific sensor can be found using the *attr -s* command, under the section "*sensor ID (index)*", it is the value given in parenthesis. If there is only one sensor configured in the RTU, then the ID is always 0.

**Example**

```
: dacq direct 0
Type <CTRL-X> to exit
```

### 4.2.9. dacq t

**Description** Used to send a command transparently to a sensor.

**Usage** dacq t <*sensor_IX*> <*cmd*> where <*sensor_IX*> is the ID of the sensor to send a command to and <cmd> is the command to be sent.

**Remarks**

Not all sensors support the transparent command. For SDI-12 <*cmd*> is a standard SDI-12 command, including the exclamation point (*!*); for Davis consoles <*cmd*> is a Davis command as described in the "Vantage Pro Serial Communication Reference Manual" (see Davis Instruments web site http://www.davisnet.com); for Thies <*cmd*> is a Thies command as described in the Thies TDL14 or DL16 User Manual.
The ID of a specific sensor can be found using the *attr -s* command, under the section "*sensor ID (index)*", it is the value given in parenthesis. If there is only one sensor configured in the RTU, then the ID is always 0.

**Example**

```
: dacq t 0 0I!
013 CSL    CS215 2.0
```

### 4.2.10. hist

**Description**

Historians command and subcommands, used to inspect the sensor data stored in the local memory (database).

**Usage**

hist [ -h | <*cmd*> ] with the *-h* parameter, the list of all subcommands will be shown.
hist <*hist#*> <*date*> *[<time>]* Display the first 20 values stored for the historian <*hist#*> starting with <*date*> and <*time*>. Without a parameter, the next 20 values (or less if not so many are available) for the previously selected historian will be displayed. The <*time*> parameter may be left out, in which case the time will be automatically set to 0:00 for the specified day.
The hist subcommands are described in the following sections.

**Remarks**

To look-up the historian ID corresponding to a certain tag, use the *hist map* command (see below).

**Example**

```
: hist 5 1/10/20
  5: 01/10/2020 00:00:00 23.540001[0]
  5: 01/10/2020 00:10:00 23.540001[0]
  5: 01/10/2020 00:20:00 23.570000[0]
[…]
  5: 01/10/2020 03:00:00 23.500000[0]
  5: 01/10/2020 03:10:00 23.490000[0]
: hist
  5: 01/10/2020 03:20:00 23.510000[0]
  5: 01/10/2020 03:30:00 23.480000[0]
  5: 01/10/2020 03:40:00 23.510000[0]
[…]
  5: 01/10/2020 06:20:00 23.410000[0]
  5: 01/10/2020 06:30:00 23.450001[0]
```

### 4.2.11. hist info

**Description**

Show the data interval stored for a specific tag.

**Usage**

hist info <*hist#*> where *hist#* is the historian for which to display the dates of the oldest and the newest record.

**Remarks**

To look-up the historian ID corresponding to a certain tag, use the *hist map* command (see below). Depending on the number of records and the number of tags in the data base, the response to this command may take up to 20 seconds.

**Example**

```
: hist info 5
Oldest: Sat May 23 19:20:00 2020
Newest: Mon Dec 14 16:50:00 2020
```

### 4.2.12. hist stat

**Description**

Used to display historian statistics.

**Usage**

hist stat.

**Remarks**

A table will be displayed with several columns. The following information is of relevance for the Metrilog support technicians, therefore it is not important to understand exactly what it means (refer to the example below):
Sensor: is the sensor index as stored in the RTU.
ID: is the sensor ID as retrieved from the M2M Gateway.
Type: is the sensor type (SDI-12, Thies, Davis, etc.).
Stream: is an internal index associating a historian with a sensor.
Block: a logical block in the storage (there are currently 768 blocks allocated to the historians).
First: date of the oldest record for the sensor.
Last: date the newest record for the sensor.

**Example**

```
: hist stat
Sensor  ID      Type    Stream  Block   First                   Last
-------------------------------------------------------------------------------------
  0     78473   SDI-12    1      727    23/05/2020 19:20:00     14/12/2020 16:50:00
  1     77178   SDI-12    0      725    23/05/2020 19:20:00     14/12/2020 16:50:00
  2     79630   Intern    4      724    23/05/2020 19:20:00     14/12/2020 16:00:00
```

### 4.2.13. hist map

**Description**

Used to display the historians map.

**Usage**

hist map.

**Remarks**

The command will display a table with the mapping of the historians to the sensor IDs, as well as their status (active/deleted). The stream to which a historian belongs will be also shown. A "deleted" historian is one that is no more active, it will be "recycled" when new sensor will be added after all historians have been used (currently 128).
Note that deleted historians do not belong to any stream.
To look-up the historian for a certain tag, first use the *attr -t* command to identify the ID of a tag (or from the M2M Gateway: select the sensor, click the *Info* tab there you will find the *Public Node ID*); using the ID, look-up the historian in the table shown by the *hist map* command.

**Example**

```
: hist map
Hist.   Tag ID  Stream  Status
------------------------------
  0      77180     0    active
  1      77179     0    active
  2      78482     1    active
  3      78481     1    active
  4      78480     1    active
  5      78479     1    active
  6      78478     1    active
  7      78477     1    active
[…]
 11      79625     –   deleted
 12      79624     –   deleted
 13      79623     –   deleted
[…]
 17      79631     4    active
 18      79632     4    active
 19      79633     4    active
 20      79955     4    active
```

## 4.3. Data Acquisition Legacy Commands

The commands in this group are maintained to assure compatibility with legacy systems. Their use is not recommended for new implementations; use a variant of the *dacq* command instead. The main difference between the equivalent *dacq* commands and the legacy commands is that for the former a valid configuration is required to access the sensors, whereas for the later it is not.

### 4.3.1. sdi t

**Description** Used to issue a command in transparent mode to an SDI-12 sensor.

**Usage** sdi t <*command*> where <*command*> is a standard SDI-12 command, including the sensor's address and the exclamation point (*!*).

**Remarks** The command implements the standard SDI-12 timing.

**Examples**

```
: sdi t AI!
A13MetrilogM512rC1.4.1055#000000

: sdi t AM!
A0048
: sdi t AD0!
A+25.17+31.71+0+0+1+0+1+12.03
```

### 4.3.2. thi t

**Description** Used to issue a command in transparent mode to a Thies data logger.

**Usage** thi t <*command*> where <*command*> is a Thies command as described in the Thies TDL14 or DL16 User Manual.

**Remarks** The command appends the SOT and EOT characters required by the Thies serial protocol, therefore the user must not add them.

**Examples**

```
: thi t XX
3.30
```

```
: thi t ZZ
17:20:12

: thi t DD
04.08.20

: thi t mm
17.3425 260  2.8  20.6  48.3   9.3  20.7 1009.6   0.0 0  249 1  502.1  25.2  0   0.0 04.08.20
17:20:33
```

### 4.3.3. thi direct

**Description** Used to directly connect to a sensor attached to the MPI connector.

**Usage** thi direct

**Remarks** This command is essentially equivalent to the *connect /dev/mpi* command. To exit the command, enter *<ctrl-X>*. To be able to use the *direct* command, the sensor/logger commands must be known. Thies loggers and Davis consoles have their own proprietary protocols, for more details consult their user manuals.

**Examples**

```
: thi direct
Type <CTRL-X> to exit

       Momentane Messwerte:
       --------------------

 Datum / Zeit:  04.08.20 17:18:56
 Windrichtung :293 Grad
 Windges. (75): 2.2 m/s
 Temperatur 2m: 20.6 Grad C
 Rel. Feuchte : 49.0 %
 Taupunkt     :  9.5 Grad C
 Temperatur5cm: 20.5 Grad C
 Luftdruck (a):1009.6 hPa
 Niederschlag :  0.0 mm
 Nied.-Melder :0
 Gl-Strahlung :  99 W/qm
 Sonne ja/nein:1
 Sonnen.-Dauer: 500.5 min/d
 Temperatur(w): 24.7 Grad C
 Batteriesp.  :13.4 V
 Luefterstrom :125 mA
 Heizungsstrom: 0.0 A
 SYNOP 1min   : 0 SYNOP
 Niedersch.LNM:  0.0 mm
 ENDE

: thi direct
Type <CTRL-X> to exit
OK
Jun  3 2013

OK
3.15
```

*Note: In the examples above, the user input is not visible, as nor the Thies logger, neither the Davis console echo the user input. In the first example the command <ctrl-B>mm<ctrl-C> was sent to a Thies DL16 logger, while in the second example the commands VER and NVER were sent to a Davis Vantage Pro console.*

## 4.4. Communications Commands

### 4.4.1. modem

**Description** Used to show the state of the built-in cellular modem.

**Usage** modem [ -h ]
The information provided may help debug cellular connection problems.

**Example**

```
: modem
Modem status:
    Operator: 23205 (automatic)
    Registered, roaming
    Network: LTE, LAC/CI: 27a5/1b21964 (10149/28449124)
    RSSI: -63dBm (25)
    Service: LTE
    Service setting: auto
    SIM1 CCID: 8943052050014004571
    SIM2 CCID: 89430103225093048876
    Modem model: SIMCOM_SIM7600G-H
    Modem version: LE20B05SIM7600G22
    IMEI: 862636056478355
```

### 4.4.2. net

**Description** Used to show the network status, download files or initiate a session to communicate with the M2M Gateway.

**Usage** net [ -h ] to show the current network status; use the *-h* option to show the command usage.
net { up | down | session } are subcommands and are described in the following sections.

**Remark** The net up and down subcommands are not available over telnet.

**Examples**

```
: net
Network status: down
: net up
: net
Network status: up
        IP address: 10.96.184.76
        DNS1: 213.33.99.70
        DNS2: 80.120.17.70
```

### 4.4.3. net up

**Description** Starts up the IP network (connect to the Internet).

**Usage** net up

**Example**

```
: net up
```

### 4.4.4. net down [-f]

**Description** Shuts down the network connection (disconnect from the Internet).

**Usage**

net down
The *-f* flag forces a disconnect if the network was started by another thread than the CLI.

**Example**

```
: net down
```

### 4.4.5. net session

**Description**

Executes a session with the M2M Gateway.

**Usage**

net session

**Remarks**

A complete session is executed; normally the sessions are automatically executed by the RTU, as specified by the *connectSchedule* attribute. This command allows the execution of an outstanding session manually, or if the *connectMode* attribute is set to 1.

**Example**

```
: net session
```

### 4.4.6. location

**Description**

Used to show location information (M717 rev D only).

**Usage**

Location { -x | -h }
The *-x* option forces reloading the Qualcomm's gpsOneXTRA data.

**Example**

```
: location -h
Usage:    location [ -x | -h ]
          the -x option forces reloading the Qualcomm's gpsOneXTRA data

: location
Please wait...
Lat/Lon: 54.324951 13.533506  deg.
Alt: 14.90 m
Satellites: 10
Date: Sun Aug 10 00:00:16 2025
```

## 4.5. Command Line Interface Error Messages

The following list describes the error numbers the CLI may return and their meaning:

| | | |
|---|---|---|
| CMD_NOT_FOUND | 1 | Invalid command |
| INVALID_PARAM | 5 | Invalid parameter |
| NOT_ALLOWED | 6 | Command not allowed from this port |
| NO_SUCH_DEVICE | 7 | The requested character device does not exist |
| MALLOC_ERROR | 8 | Error allocating memory (out of memory) |
| FILE_SYSTEM_ERROR | 9 | Generic file system error |
| SERIAL_SETATTRIB_ERR | 10 | Serial port error |
| USR_TIMEOUT | 11 | User timeout (no keys pressed for a defined time) |
| WRONG_PASSWORD | 12 | Invalid password |
| UNEXPECTED_ANSWER | 13 | Unexpected answer (e.g. from a sensor) |
| FLASH_ERROR | 31 | Error writing to/reading from the flash memory |
| EXITCOMMAND | 99 | User requested exit from CLI |

# 5. Attributes

This section details the attributes used to define the RTU functionality. It is important first to understand the function of a certain attribute before attempting to change its value, as improper manipulation of the attributes can lead to malfunction. Attribute's values may be changed through the M2M Gateway User Interface.

It is not possible to directly add or delete attributes to/from an existing node. This may be done only by applying a different template to the respective node (use the *Apply Template* tab on the M2M Gateway). If the required template does not exist, you can:

- Create a new template e.g. by using the *Save as Template* tab on the existing node.
- Add or delete attributes to/from the newly saved template.
- Use the *Apply Template* tab on the initial node.

(See also the section on how to create new templates).

There are three levels of attributes: RTU, sensor and tag.

The description below takes as reference the attributes view on the M2M Gateway, but explains also the RTU internal attributes. Which attributes are displayed on the M2M Gateway largely depends on the user's rights while on the CLI it depends on the user type (standard or superuser). Generally, there are more attributes visible on the RTU than on the M2M Gateway, however, they are shown only as information.

*Note: Not all attributes are described in this manual, rather only those available to realm administrators on the M2M Gateway.*

## 5.1. RTU Attributes

| Attribute | Description |
|---|---|
| assignedIP | The IP address assigned by the cellular provider while the IP connection is up. With the appropriate equipment it can be used to telnet into the RTU (see also the *cliOverIP* attribute). |
| cliOverIP | If true, at the next IP session a listening telnet server on the port defined by the *cliPort* attribute will be started; if no client connect to the server after a number of seconds defined by the *listenTimeout* attribute, the server will shut down. This attribute defaults to *false* and will automatically be set to *false* after a session ends or the server times out. |
| cliPasswd | CLI password; the password is autogenerated by the RTU and sent to the M2M Gateway. |
| cliPort | The port number on which the telnet server will listen (default 22). |

| Attribute | Description |
|---|---|
| connectSchedule | Defines the connection schedule of the RTU. It uses a *cron* syntax, ex: **/10* means that the RTU should connect every ten minutes, starting with minute 0. A complete *cron* string may specify minutes, hours, days, months and the day of the week. The smallest unit that can be set is therefore one minute. However, for practical reasons related to cellular communication specifics, the minimum recommended interval is five minutes. For additional details on configuring *cron*, consult the unix man pages for *crontab (5) (*https://man.openbsd.org/crontab.5*)*. |
| connectSpread | The value of this attribute is specified in seconds. A random value in the interval 0 ... *connectSpread* will be added to the computed *connectSchedule* (default 30). If 0, no connection randomising will be used. This is used to distribute the load on the server when many RTUs initiate a connection at the same time. |
| date | Time stamp of the last successful update on the server; it represents the time stamp for all volatile attributes. |
| gsmCCID | CCID of the SIM card. |
| gsmCCIDAlt | CCID of SIM2 card (if available, and only on M717 rev D devices) |
| gsmIMEI | IMEI of the cellular modem. |
| gsmNumber | Own number (MSISDN) allocated to the RTU (i.e., SIM card). Only the number of the active SIM card is shown (gsmCCID). |
| gsmType | The cellular modem type. |
| gsmVersion | The firmware version of the cellular modem. |
| hwID | Serial ID of the RTU. |
| hwType | RTU type. |
| manufacturer | RTU manufacturer's name. |
| mpiBaudRate | The baud rate used by the MPI (the 8-pin connector, default 19200 Baud). |
| mpiMode | The operation mode of the MPI (default RS-232); it can be:<br>– RS-232<br>– RS-422<br>– RS-485<br>For the wiring of the MPI connector see also the section Installation. |
| mpiTermination | If true, it activates an 120Ω terminating resistor (valid only in RS-422 and RS-485 modes; default *false*). |

| Attribute | Description |
|---|---|
| rtcUpdateSchedule | Defines the schedule used to update the Real Time Clock (RTC); this is normally set to once a day at midnight ("0 0"). It uses the standard *cron* format. For additional details on configuring *cron*, consult the unix man pages for *crontab (5) (https://man.openbsd.org/crontab.5)*. |
| sdiBaudRate | The baud rate used by the SDI (the 4-pin connector, default 1200 Baud). |
| sdiMode | The operation mode of the SDI (default SDI-12); it can be:<br>– SDI-12<br>– RS-485<br>– CAN<br>For the wiring of the SDI connector see also the section Installation. |
| sdiMsg | Send SDI command(s) in pseudo-transparent mode to an SDI sensor. The attribute is intended to be updated through the M2M Gateway. More than one command can be sent in one string; while spaces between commands are optional, they improve readability. If a delay is required between commands, it can be specified as \*n*, where *n* specifies the delay in seconds. Example: *0M! \5 0D0!* The results returned by the sensor(s) are piped to the log. |
| sdiTermination | If *true*, it activates an 120Ω terminating resistor (valid only in RS-485 and CAN modes; default *false*). |
| swBootVersion | The version of the RTU's bootloader. |
| swVersion | The version of the RTU's firmware. |
| telemetry | A list of telemetry values reflecting various RTU conditions (power supply, microcontroller's temperature, RSSI, etc.). |
| timeZone | The ISO definition of the time zone where the RTU operates (default *UTC*). The *timeZoneOffset* attribute is derived form this attribute. |
| uptime | Time since the last RTU's reboot. |

## 5.2. Sensor Attributes

| Attribute | Description |
|---|---|
| acquisitionMode | Defines how the sensor will be sampled (default 2). Following modes are defined:<br>0 – no sampling, sensor is disabled.<br>1 – not used.<br>2 – automatic mode, defined by the sensor's *aquisitionSchedule* attribute. |

| Attribute | Description |
|---|---|
| acquisitionSchedule | Defines the sensor sampling schedule. It uses a *cron* syntax, ex: **/10* means that the sensor should be sampled every ten minutes starting with minute 0. A complete *cron* string may specify minutes, hours, days, months and the day of the week. The smallest unit that can be set is therefore one minute. For additional details on configuring *cron*, consult the unix man pages for *crontab (5) (*https://man.openbsd.org/crontab.5*)*. |
| archiveInterval | Specifies the archiving interval in minutes (default 10) on a Davis WeatherLink logger/interface. This attribute is valid for Davis loggers only. |
| davisVersion | Returns the software version of the Davis WeatherLink logger/interface. This attribute is valid for Davis loggers only. |
| iconName | The icon graphic that will be used to display the sensor on the M2M Gateway (default *COMBO*); the graphic must already exist on the server (on the UI of the M2M Gateway, click the *Info* button, then *Icons*). Note: this attribute is used only by the M2M Gateway and it does not exist in the RTU. |
| lastDate | The date the sensor was sampled last time. |
| putdataDate | The date of the sensor's last values pushed unto the server (M2M Gateway). Normally the equivalent RTU attribute is used, unless a sensor has a different schedule. This is an RTU internal attribute, it does not exist on the server. |
| sdiAddress | The SDI sensor address. This attribute is valid for SDI-12 sensors only. |
| sdiInfo | Returns the result of the "*I*" (Info) command to an SDI-12 sensor. This attribute is valid for SDI-12 sensors only. |
| sdiMethod | The acquisition method used; as per the SDI-12 specification, this can be *M*, *C*, *R*, *MC*, *CC*, *RC* and *V*. This attribute is valid for SDI-12 sensors only. |
| sdiPostmethod | An (optional) SDI-12 command sent after sampling the sensor. Usually this is an SDI-12 *X* command. The address and the SDI-12 terminator (*!*) should be skipped (the address is defined by the *sdiAddress* attribute). This attribute is valid for SDI-12 sensors only. |
| thiesInfo | Returns the software version of the Thies data logger. This attribute is valid for Thies TDL14 and DL16 data loggers only. |
| thiesMethod | Defines the request sent to the Thies data-logger. Currently it can only be *ds*. This attribute is valid for Thies TDL14 and DL16 data loggers only. |

## 5.3. Tag Attributes

| Attribute | Description |
|---|---|
| EUID | Engineering Unit Identification. Each tag has a value type (e.g., mm, ºC, km/h, etc.), which in turn has an ID assigned. The EUID can be read by a data retrieving client (e.g., over the addUPI protocol) and display the tag values with the proper type. A list of all defined EUIDs can be reached over the M2M Gateway UI: click the *Info* button, then the *Engineering Units* tab. This attribute does not exist on the RTU. |
| iconName | The icon graphic that will be used to display the sensor on the M2M Gateway; the graphic must already exist on the server (on the UI of the M2M Gateway, click the *Info* button, then *Icons*). Note: this attribute is used only by the M2M Gateway and it does not exist in the RTU. |
| commandMode | Defines how commands defined by the *sdiCommand* attribute will be issued to output tags (default 2). Following modes are defined:<br>0 – no commands, tag is disabled.<br>1 – not used.<br>2 – commands will be sent in automatic mode, defined by the *commandSchedule* attribute.<br>This attribute is valid for SDI-12 actuators only. |
| commandSchedule | Defines the schedule to send commands defined by the *sdiCommand* attribute to an output tag. It uses a *cron* syntax, ex: *0 1-5* means that the command will be sent between one and 5 AM, every hour at minute 0. A complete *cron* string may specify minutes, hours, days, months and the day of the week. The smallest unit is therefore one minute. For additional details on configuring *cron*, consult the unix man pages for *crontab (5) (*[https://man.openbsd.org/crontab.5](https://man.openbsd.org/crontab.5)*)*. This attribute is valid for SDI-12 actuators only. |
| conversionParams | A string representing a first or second degree polynomial that is used by the RTU to convert the input value; the converted value will be stored in the internal memory. This feature can be used to convert e.g. Fahrenheit degrees to Celsius, or vice-versa. As an example, the formula Celsius = 0,5555*Fahrenheit - 17,7778 (generic:  y = a*x + b) translates to the string 0.5555 -17.7777. For second degree polynomials, three parameters are required, and their order is always from left to right (ex.: a, or a b, or a b c). This attribute is valid for Davis loggers only. |

| Attribute | Description |
|---|---|
| dashValue | Value returned by the Davis WeatherLink that represents an invalid tag value. When a tag returns a value equal to the *dashValue* attribute, a status 1 (missing data) is generated for this value. This attribute is valid for Davis loggers only. |
| davisIndex | Defines the position of the tag's value in the string returned by a *DMPAFT* command. The *davisIndex* starts with 0, for the first position in the string. This attribute is valid for Davis loggers only. |
| davisSize | Defines the value's length (in bytes) in the string returned by the *DMPAFT* command. This attribute is valid for Davis loggers only. |
| lastValue | The last value returned by the tag. On the server this attribute does not exist, as its values are stored in the historians. |
| putdataDate | The date of the sensor's last values pushed unto the server (M2M Gateway). Normally the equivalent RTU or sensor attribute is used, unless a tag has a different schedule, or if the data flow to the server was interrupted. This is an RTU internal attribute, it does not exist on the server. |
| sdiCommand | The string representing the command to be sent to an output tag; usually it is an *X* command (or a manufacturer custom command). The address and the SDI-12 terminator (*!*) should be skipped (the address is defined by the *sdiAddress* attribute). This attribute is valid for SDI-12 actuators only. See also the *commandSchedule* and *commandMode* attributes. |
| sdiIndex | Defines the position of the tag's value in the string returned by a *Dn* or *Rn* command. The *sdiIndex* starts from 0, for the first position in the string. This attribute is valid for SDI-12 sensors only. |
| status | The status of the tag after being sampled. 00 means OK, other values signify errors (e.g., 01 "missing data"). On the server this attribute does not exist, its values are stored in the historians. |
| thiesIndex | Defines the position of the tag's value in the string returned by a Thies datalogger *ds* command. The *thiesIndex* starts from 0, for the first position in the string. This attribute is valid for Thies TDL14 and DL16 data loggers only. |
| value | Not used. |

# 6. Technical Specifications

| Parameter | Value |
|---|---|
| Cellular modem | GPRS/EDGE/WCDMA/LTE Cat 4 |
| Interfaces | RS-232, RS-422, RS-485, CAN, SDI-12, USB 2.0 |
| Supported protocols | TCP/IP, HTTP, HTTPS |
| Internal storage | 16 Mbytes, non-volatile |
| Sensor sampling interval | Programmable (minutes, hours, days, weeks) |
| Communication interval | Programmable (minutes, hours, days, weeks) |
| Power supply | External 6 to 30 Volt (M12 connectors); 5 Volt (USB) |
| Power consumption | Standby, receive enabled: typ. 2.8 mA (at 12 V)<br>Standby, receive disabled: typ 0.8 mA (at 12 V)<br>Transmit: max. 300 mA (at 12 V), max. 600 mA (at 6 V) |
| Operating temperature | -30ºC to +60ºC |
| Dimensions | 110/75/55 mm |
| Weight | 400 g |
| Environmental protection class | IP66 |