HTTP Data Logging Protocol

Version 1.5

(introduced with Meteohub 4.9)

1. Mission Statement

The mission of this document is to provide a protocol specification that allows to handover weather data logged by Meteohub to WD and other PC weather programs. The underlying transport layer should be TCP/IP as this allows to connect the data logger from anywhere in the world as long as logger and PC program do have Internet access. The logger should be able to provide live data and also archived data on request. The protocol should be able to handle all types of sensors of current weather stations.

2. Basic Assumptions

- Communication should be realized as HTTP requests (GET and POST flavor should be supported). This has severe advantages compared to other TCP/IP-based protocols:
  - going through firewalls is very easy and widely supported
  - HTTP is supported on all known platforms, in all programming languages and development environments
  - the logger can be easily tested by simple browser requests
  - request/receive transactional logic behind HTTP requests fits the purpose
- The logger should provide two separate modes.
  - "INFO" mode for information about connected sensors, station data, etc.
  - "DATA" mode for retrieving logged data for a defined time frame or most recent data on all sensors
- The logger should reply data wrapped as TEXT/PLAIN in one of two types (the requesting program decides on the type)
  - flat text, line-based (each response is terminated by a line feed).
  - XML (enclosed by a "<logger> ... </logger>" tag)
  - By setting URL parameter "quotes" to "1" a strict XML format will be generated, with all values in quotes and "/>" as closing bracket. This change has not been generally applied, because WD will no longer be feedable using this strict format.

3. Defined Generic Sensors

Meteohub supports a broad range and number of sensors. Regardless what weather station is connected Meteohub represents the sensor data by a set of defined sensor types. A specific sensor corresponds to one of the defined sensor types and is characterized by an unique ID that is composed from the type name and a number. Most sensor types can have 20 specific sensors (0...19) the thermo, thermo-hygro and data type support up to 40 sensors (0...39).

Sensor type	Sensor ID	Values	Comment
THB: thermo-hygro-baro	thb0, thb1, ... , thb19	1. temperature [°C] 2. humidity [%] 3. dew point in [°C] 4. station pressure [hPa} 5. sealevel pressure [hPa] 6. simple forecast [num[1]]	this is mostly an indoor sensor, often directly integrated into the weather station. Normally just one sensor available, named thb0. Humidity and forecast are integer, other values have 1 decimal
TH: thermo-hygro	th0, th1, ... , th39	1. temperature in °C 2. humidity in % 3. dew point in °C	th0 is the standard outdoor sensors, other sensors can be outdoor or indoor, 1 decimal
T: thermo	t0, t1, ... , t39	1. temperature in °C	pure temperature sensor without humidity information, 1 decimal
WIND: wind	wind0, wind1, ..., wind19	1. direction[2] [°] 2. gust speed [m/s] 3. aver. wind speed [m/s] 4. wind chill[3] [°C]	direction is integer, other values have 1 decimal
RAIN: rain	rain0, rain1, ... , rain19	1. rain rate [mm/h] 2. rain fall total [mm] 3. delta rain fall [mm] since last request[4]	total rain is counted since beginning of sensor's recording, 1 decimal
UV: uv index	uv0, uv1, ... , uv19	1. uv index [uvi]	1 decimal
SOL: solar radiation	sol0, sol1, ... , sol19	1. radiation [W/m^2]	no decimals
DATA: numerical values	data0, data1, ... , data39	1. numerical value	any value, 2 decimals

Examples

- "thb0 22.5 55 5.2 1004.7 1007.9 3"
  indicates that indoor temp in 22.5°C, 55% relative humidity, dew point at 5.2°C, pressure at station level is 1004.7 hPa, pressure computed to sea level is 1007.9 hPa, forecast is: sunny
- "wind0 45 5.3 3.1 7.3"
  states that wind is blowing from NE (45°), with gust speed 5.4 m/s, average wind speed 3.1 m/s, wind chill is 7.3 °C (measured against temperature reading of th0)
- "rain0 1.5 4536.0"
  states that we have rain at a rate of 1.5 mm per hour, total rain fall measured so far by the sensor is 4536 mm. Values in mm are equivalent to liter per m^2.

[1] num is a number that represents a very basic forecast: 0 = rainy, 1 = cloudy, 2 = some clouds, 3 = sunny, 4 = snowy, 5 = clouds at night, 6 = clear night

[2] 0° = North, 90° = East, 180° = South, 270° = West

[3] based on temperature reading of th0

[4] delta gets only reported in requests without start/end time stamp. delta gives amount of rain since last request

4. Commands to be supported by the Logger

The data logger acts on HTTP requests. parameters can be given as URL parameters (GET mode) or by a transferred data record with name-value pairs (POST mode). The PC weather program has freedom to choose.

4.1 INFO Requests

These requests as for general information about the connected station, location on earth, timezone, and available sensors.

Request Parameter	Flat Text Response	XML Response	Comment
type=xml	none, flat text responses will separated by line feeds	none, XML resonses will be enclosed in "<logger> ... </logger>" tags.	this switches to XML response mode. Without this parameter plain text will be reported.
quotes=1	none	none, will use strict XML syntax for output with all values in quotes and "/>" as closing tag bracket.	introduced to allow for strict XML parsing, not generally applied to keep compatibility to WD feed definition.
mode=info	none	none	tells the logger that this is an "info request"
info=station	WMR-928 WMR-100 WMR-200 TE-923 VANTAGE RFXCOM	<station>WMR-928</station> <station>WMR-100</station> <station>WMR-200</station> <station>TE-923</station> <station>VANTAGE</station> <station>RFXCOM</station>	connected weather station
info=version	Meteohub x.x	<version>Meteohub x.x </version>	used version of Meteohub

info=utcdate	YYYYMMDDHHMMSS	<utcdate
		year="YYYY"
		month="MM"
		day="DD"
		hour="HH"
		minute="MM"
		second="SS">
		YYYYMMDDHHMMSS
		</utcdate>
			date/time in UTC

info=localdate	YYYYMMDDHHMMSS	<localdate
		year="YYYY"
		month="MM"
		day="DD"
		hour="HH"
		minute="MM"
		second="SS">
		YYYYMMDDHHMMSS
		</localdate>
			date/time in local time

info=latitude	DEG,MIN,SEC,HEMI	<latitude
		degree="DEG"
		minute="MIN"
		second="SEC"
		hemisphere="HEMI">DEG, MIN,SEC,HEMI</latitude>
			hemisphere can be "N" (north) or "S" (south)

info=longitude	DEB,MIN,SEC,MERID	<longitude degree="DEG"	meridian can be "E" (east) or "W" (west)

Request Parameter	Flat Text Response	XML Response	Comment
		minute="MIN" second="SEC" meridian="MERID">DEG, MIN,SEC,Merid</longitude>	
info=altitude	ALT	<altitude>ALT</altitude>	altitude above sea level in meters
info=sensorids	SENS1,SENS2,...,SENSn	<sensor id="SENS1" type="SENSTYPE1"> SENSNAME1 </sensor> ...	"SENSx" is sensor ID (see Section 3. "SENSTYPEx" is sensor type (see section 3) SENSNAMEx is a speaking nick name the user has given the sensor inside Meteohub.
info=sensortypes	SENSTYPE1,SENSTYPE2,...,SENSTYPEn	no separate tag, included in "sensor"	see above
info=sensornames	SENSNAME1,SENSNAME2,...,SENSNAMEn	no separate tag, included in "sensor"	sea above

Examples

- Request: "http://..../meteolog.cgi?type=xml&quotes=1&mode=info&info=station&info=utcdate&info=sensorids"
  Response:

```
<logger>
  <station>WMR-928</station>
  <utcdate year="2008" month="2" day="27" hour="18" minute="35"
           second="9">20080227183509</utcdate>
  <sensor id="thb0" type="THB">indoor (living room)</sensor>
  <sensor id="th0" type="TH">outdoor (car port)</sensor>
  <sensor id="wind0" type="WIND">wind</sensor>
  <sensor id="rain0" type="RAIN">rain (backyard)</sensor>
  <sensor id="uv0" type="UV">uv sensor</sensor>
</logger>
```

- Request: "http://..../meteolog.cgi?mode=info&info=station&info=utcdate&info=sensorids&info=sensortypes& info=sensornames"
  Response:

```
WMR-928
20080227183509
thb0,th0,wind0,rain0,uv0
THB,TH,WIND,RAIN,UV
indoor (living room),outdoor (car port),wind,rain (backyard),uv sensor
```

4.2 DATA Requests

Data requests ask the logger to report all data records that have a time stamp between boundaries given by a start and end time stamp. This allows retrieval of data from specified time frames in the past as well as reading the most actual data in a live mode.

To realize an ongoing incremental reading of actual sensor data, the following flow of control should be realized by the PC program. Incremental read can be easily realized by taking the date stamp of the last received record as start parameter for the next request. This incremental approach allows the logger to optimize its performance by keeping in mind where delivery of data to the PC program has stopped. Having this information cached improves performance of incremental data delivery dramatically. On the other hand it is very easy for the PC program to ask for next data records by using the the last received time stamp as the start value for the next request.

The flow of control for ongoing reading of live date can be like this:

1. PC requests the last logged weather data record ("mode=data", no start/end parameter).
2. Logger returns last values of all logged weather data sensors. Notice newest time stamp received by this request (time stamp has format: "YYYYMMDDHHMMSS").
3. PC evaluates received data. After having done this PC might wait a few seconds and then asks the logger to report data with a newer time stamp than the newest data just received. As the logger doesn't remember time stamps of delivered data the PC program includes the time stamp of the last received data into the request and leaves end time stamp empty. ("mode=data&start=YYYYMMDDHHMMSS").
4. Logger returns all data records newer than the given time stamp. If it is more than 100 records, just the oldest 500[5] are reported. If there is no data available past the given time stamp, an empty response is returned. Continue at step 3.

As an alternative you can also just send a request without start/end parameter every couple of minutes. Disadvantage of this method is that you get duplicated data that has not updated since last request and that you will not get notice of data that has changed more than once since last request. This approach to read the data logger is less accurate (duplicates, data losses) and by it's need for fast polling it adds stress to the system, but it might be more easy to implement.

Request Parameter	Flat Text Response	XML Response	Comment
type=xml	none, flat text responses will separated by line feeds	none, XML responses will be enclosed in "<logger> ... </logger>" tags.	this switches to XML response mode. Without this parameter plain text will be reported.
quotes=1	none	none, will use strict XML syntax for output with all values in quotes and "/>" as closing tag bracket.	introduced to allow for strict XML parsing, not generally applied to keep compatibility to feed WD.
sensor=string	none	none	tells the logger only to report data of sensors that match "string". Sensor parameter can be specified multiple times in URL[6].
"start" parameter omitted	last data record before reaching the time	last data record before reaching the time stamp given	useful to get a time stamp to start with, which is

[5] If the limit is reached and there are more data records with the same time stamp these are also reported. Therefore, the result can be a bit more than 500 records. This is especially true for weather stations like Vantage or TE923 which report the whole sensor array at the same time.

[6] string is used for a substring search in the data line. "th" matches all lines that have "th" in it. This will selct sensors like "thb0", "th0", "th1". "th1" selects sensor "th1" but also "th10". "th1+" matches just "th1", not "th10".

Request Parameter	Flat Text Response	XML Response	Comment
	stamp given by "end"	by "end"	direct predecessor to the times tamp given by "end" parameter .
start=YYYYMMDDHHMMSS	max. 500 data records, one sensor per line	max. 500 data records in sensor type specific tags	requests data with time stamp newer than YYYYMMDDHHMMSS
"end" parameter omitted	max. 500 data records, one sensor per line	max. 500 data records in sensor type specific tags	requests data up to last recently logged ones
end=YYYYMMDDHHMMSS	max. 500 data records, one sensor per line	max. 500 data records in sensor type specific tags	requests data with time stamp equal or older than YYYYMMDDHHMMSS
"start" and "end" are omitted	latest set of data records for all sensors	latest set of data records in sensor type specific tags	this is in a live data reading context

The logger reports data based a sensor's type for each of the sensor readings in scope. The following table explains how the response for each type of sensor looks like.

Sensor Type	Flat Text Response	XML Response (quotes=1)	Comment

THB	timestamp[7] thb?[8] values[9]	<THB date="YYYYMMDDHHMMSS"
		id="thb?"
		temp="value"
		hum="value"
		dew="value"
		press="value"
		seapress="value"
		fc="value" />
			values as defined in section 3

TH	timestamp th? values	<TH date="YYYYMMDDHHMMSS"
		id="th?"
		temp="value"
		hum="value"
		dew="value" />
			values as defined in section 3

T	timestamp t? values	<T date="YYYYMMDDHHMMSS"
		id="t?"
		temp="value" />
			values as defined in section 3

WIND	timestamp wind? values	<WIND
		date="YYYYMMDDHHMMSS"
		id="wind?"
		dir="value"
		gust="value"
		wind="value"
		chill="value" />
			values as defined in section 3

RAIN	timestamp rain? values	<RAIN
		date="YYYYMMDDHHMMSS"
		id="rain?"
		rate="value"
		total="value"
		delta="value" />
			values as defined in section 3

UV	timestamp uv? values	<UV date="YYYYMMDDHHMMSS" id="uv?"	values as defined in section 3

[7] in UTC, format: YYYYMMDDHHMMSS

[8] ? represents a number between 0 to 9 (0 to 19 in case of TH sensor)

[9] list of values as described in section 3

Sensor Type	Flat Text Response	XML Response (quotes=1)	Comment
		index="value" />	
SOL	timestamp sol? values	<SOL date="YYYYMMDDHHMMSS" id="sol?" "rad=value" />	values as defined in section 3

DATA	timestamp data? values	<DATA
		date="YYYYMMDDHHMMSS"
		id="data?"
		value="value" />
			values as defined in section 3

As each weather station has a different configuration of sensors. The following conventions how to map sensors to unique sensor IDs is used by the logger:

Weather Station	Mapping of Sensors to IDs		Comment
WMR-928, WMR-968	thb0 th0 th1...3 t1...3 wind0 rain0	indoor temp/hygro/baro sensor primary outdoor temp/hygro sensor additional temp/hygro sensors additional temp sensor wind sensor rain sensor	additional temp and temp/hygro sensors share the channel id. As a result each of the channels 1...3 can only have a T or TH sensor (not both). Therefore, number of additional sensors is limited to 3.
WMR-100	thb0 th0 th1...10 wind0 rain0 uv0	temp/hygro/baro in the base station primary outdoor temp/hygro sensor additional temp/hygro sensors wind sensor rain sensor uv sensor	
WMR-200	thb0 th0 th1...10 wind0 rain0 uv0	temp/hygro/baro in the base station primary outdoor temp/hygro sensor additional temp/hygro sensors wind sensor rain sensor uv sensor	
TE-923 (Mebus, Irox, Nexus, Honywell, ...)	thb0 th0 th1...5 uv0 wind0 rain0	temp/hygro/baro in the base station primary outdoor thermo/hygro sensor additional thermo/hygro sensors additional uv sensor wind sensor rain sensor	
Vantage	thb0 th0 th1...7 th10...13 th15...18 uv0 sol0 wind0 rain0	temp/hygro/baro in the base station primary outdoor thermo/hygro sensor additional thermo/hygro sensors additional soil thermo/hygro sensor additional leaf thermo/hygro sensor additional uv sensor solar sensor wind sensor rain sensor	additional leaf and soil sensors deliver humidity not as relative humidity measured in percent, but as an integer number. soil delivers a "centibar" value and leaf a "wetness" value.
RFXCOM	thb0...19 th0 th1...39 t1...19 uv0...19 wind0...19 rain0...19	indoor temp/hygro/baro sensor primary outdor temp/hygro sensor temp/hygro sensor temp sensor uv sensor wind sensor rain sensor	RFXCOM receiver module reads rf signals of most Oregon sensors. Therefore no hard limit in number of sensors is applicable. However, Meteohub limits number of sensors per type.

Examples

- Data Request: "http://..../meteolog.cgi?type=xml&quotes=1&mode=data&start=20080227183500"
  Response:
  ```
  <logger>
    <THB  date="20080227183508"
          id="thb0" temp="22.3" hum="63" dew="6.8" press="998.7"
          seapress=1004.3 fc="3" />
    <TH   date="20080227183510" id="th0" temp="12.5" hum="34" dew="2.3" />
    <WIND date="20080227183510" id="wind0" dir="237" gust="12.4"
          wind="8.5" chill="8.9" />
    <UV   date="20080227183521" id="uv0" index="1.2" />
  </logger>
  ```
- Data Request: "http://..../meteolog.cgi?mode=data&start=20080227183500"
  Response:
  ```
    20080227183508 thb0 22.3 63 6.8 998.7 1004.3 3
    20080227183510 th0 12.5 34 2.3
    20080227183510 wind0 237 12.4 8.5 8.9
    20080227183521 uv0 1.2
  ```
- Data Request: "http://..../meteolog.cgi?mode=data&start=20080227183500&sensor=th&sensor=uv"
  Response:
  ```
    20080227183508 thb0 22.3 63 6.8 998.7 1004.3 3
    20080227183510 th0 12.5 34 2.3
    20080227183521 uv0 1.2
  ```
- Initial Data Request: "http://..../meteolog.cgi?mode=data"
  Response:
  ```
    20080227183524 rain0 1.5 2847.5 0.0
  ```

If no records are available for a given time frame an empty record is returned ("<logger></logger>" phrase in case of xml).

4.3 Error Handling

If an error occurs the logger returns a HTML page that starts with "Error" (directly following the necessary HTML content type header). The explanation what has gone wrong is given in clear text. No data gets transmitted in case of an error.