Pogoda.cc
Meteo stations system

REST Web Service specification

Matuesz Lubecki
tel: +48 660 43 44 46
email: mateusz.lubecki@outlook.com
sp8ebc@gmail.com

State as for April 2021

Table of Contents

Preface

Weather stations avaliable through a Web Service described in this document use APRS network (Automatic Packet/Position Reporting System) for data transmission. APRS radio network works on 144.800MHz with baudrate of 1200bps using FM/AFSK modulation (ITU radio emission classifier: F2D) and HDCL / AX25 protocol stack. Simplified network structure is briefly described on Figure 1.

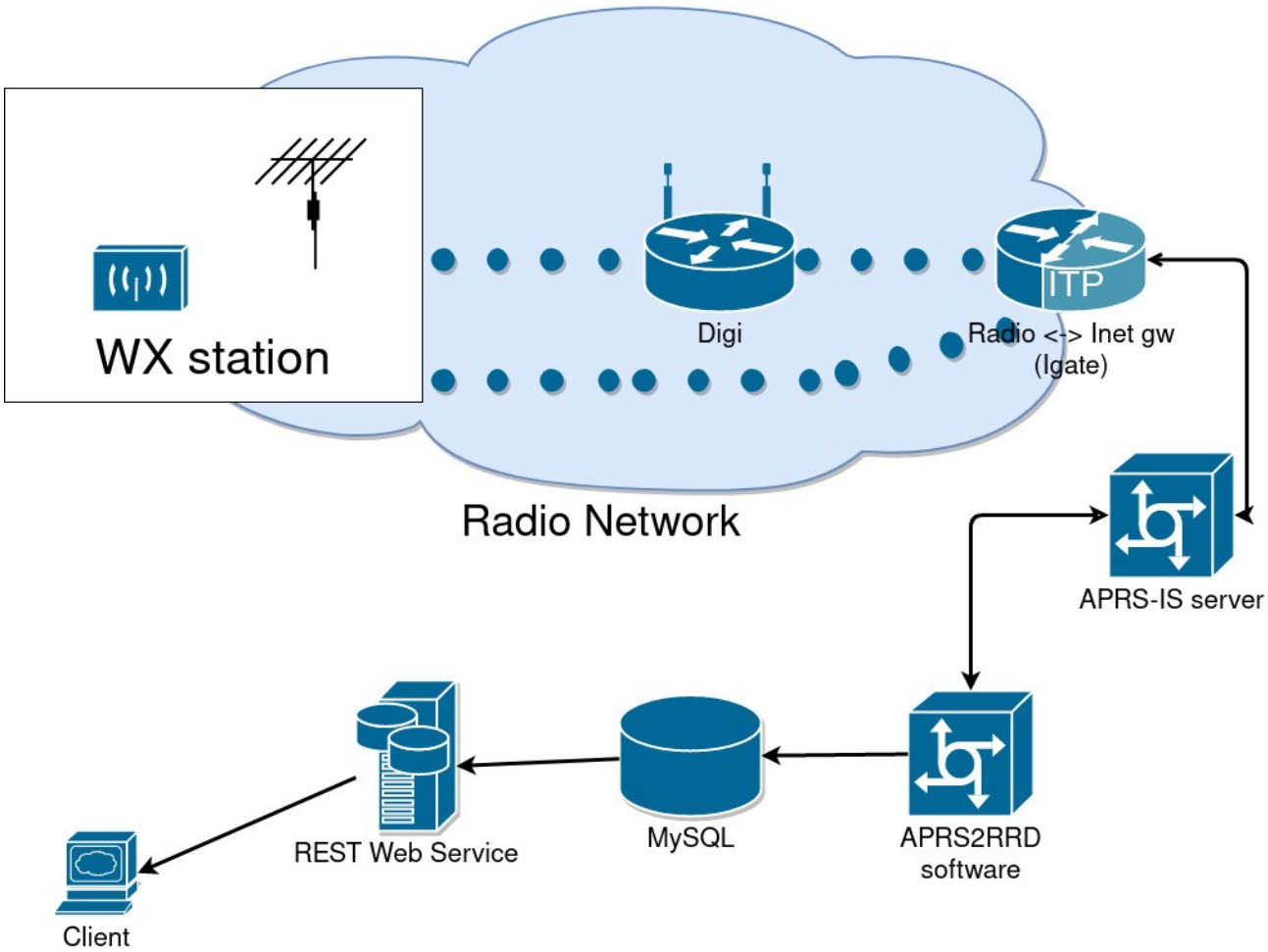

Figure 1: Simplified data flow diagram

Main assumptions:

- APRS network transmits data for instantaneous usage. Natively it does NOT persist data in any way. In theory it is possible to send a meteo data request to the station using the radio channel, but as for now such functionality is not implemented in any version of the ParaTNC firmware.
- Data could be acquisitioned directly from the radio network by listening to 144.800MHz using appropriate radio modem.
- Data could be also retrieved directly using one of many APRS-IS server by establishing a TCP connection and using standard, APRS-IS text based comm protocol. Please bear in mind that APRS-IS servers network works on p2p

principle. Servers transmits data between themselves and connected clients according to filter definitions sent during an authorization.
- Pogoda.cc service and system consist only weather stations based on ParaTNC controller. In practice any station which transmits data using standard APRS protocol and a WX format described there could be connected.

This paper describes REST WS endpoints serviced by pogoda.cc system. They servers data gathered by APRS2RRD software (https://github.com/SP8EBC/aprs2rrd-se) from APRS-IS network.

General information about REST WebService

1. Base URL: **http://pogoda.cc:8080/meteo_backend/**
2. Encoding: UTF-8
3. Response format: JSON
4. Units: SI (wind in m/s, pressure in hPa, temperature degrees C)

The web service is designed as a backed for mobile application 'pogoda.cc'. Because of that some fields of responses are not strictly related to any weather data, but rather to the application and its interface itself.

The web service is implemented in Java 8 SE language using Spring MVC framework. Wherever documentation specifies data type, this type reflect to such used in Java implementation.

GET /listOfAllStations

This web service returns the list of all WX stations defined in the system.

URI: http://pogoda.cc:8080/meteo_backend/listOfAllStations
Method: GET
Parameters: Brak
Response: An array 'stations' of StationDefinition class
Example Response:


```
{
  "stations": [
    {
      "id": 1,
      "name": "skrzyczne",
      "enabled": true,
      "callsign": "SR9NSK",
      "ssid": 2,
      "displayedName": "Skrzyczne",
      "displayedLocation": "Góra Skrzyczne, Beskid Śląski",
      "sponsorUrl": "http://skrzyczne.pogoda.cc/",
      "backgroundJpg": "http://pogoda.cc/_pogoda_cc_thumb/unnamed.jpeg",
```

```
      "backgroundJpgAlign": 5,
      "stationNameTextColour": -1,
      "moreInfo": "ParaTNC firmware DF15-04042021 by SP8EBC",
      "lat": 49.6852,
      "lon": 19.0318,
      "timezone": "Europe/Warsaw ",
      "hasWind": true,
      "hasQnh": true,
      "hasHumidity": false,
      "hasRain": false,
      "telemetryVersion": 0
    },
    {
      "id": 2,
      "name": "bezmiechowa",
      "enabled": true,
      "callsign": "SR8WXB",
      "ssid": 1,
      "displayedName": "Bezmiechowa Górna",
      "displayedLocation": "Akademicki Ośrodek Szybowcowy",
      "sponsorUrl": "http://aos-bezmiechowa.pl/",
      "backgroundJpg": "http://pogoda.cc/_pogoda_cc_thumb/bezmiechowa.JPG",
      "backgroundJpgAlign": 5,
      "stationNameTextColour": -1,
      "moreInfo": "ParaTNC firmware DF12-26022021 by SP8EBC",
      "lat": 49.5218,
      "lon": 22.4113,
      "timezone": "Europe/Warsaw ",
      "hasWind": true,
      "hasQnh": true,
      "hasHumidity": false,
      "hasRain": false,
      "telemetryVersion": 1
    },  (...) ]

}
```


Fields of ‘stations’ array element (StationDefinition class):

Name	**Type**	**Description**
id	int	Integer identifier (auto_increment)
name	String	Text identifier used then in different endpoints
enabled	Boolean	Controls if station is shown on ‘All Stations’ list in pogoda.cc application. It DOES NOT mean if the station itself works or not.
Callsign	String	Callsign in APRS radio network
SSID	int	SSID in APRS radio network
displayedName	String	Station name displayed on ‘All Stations’ list in pogoda.cc
displayedLocation	String	Station localization displayed in ‘Station Details’
sponsorURL	String	Clickable URL displayed in ‘Station Details’ pogoda.cc application
backgroundJpg	String	URL to graphic / photo used a background in ‘Station Details’
backgroundJpgAlign	int	Align of background. A parameter of `topBackground.setScaleType(…)` in `protected void onCreate(Bundle savedInstanceState) { … }` in `cc.pogoda.mobile.pogodacc.activity.StationDetailsActivity`

stationNameTextColour	int	Text color of displayedName. A parameter of `stationName.setTextColor(station.getStationNameTextColor());`
moreInfo	String	Text field displayed in ‘Station Details’ activity
lat	float	Station coordinates
lon	float	Station coordinates
timezone	String	Timezone used by the station
hasWind	boolean	If this station currently measures wind or not
hasQnh	boolean	If this station currently measures barometric pressure
hasHumidity	boolean	If this station currently measures humidity
hasRain	boolean	If this station currently measures rain
telemetryVersion	int	Parameter exclusive for ParaTNC controller and firmware. It selects a format of telemetry packets send by the ParaTNC.

GET /summary?station={station_name}

Returns a summary of most recent data sent by the stations. The summary is generated from data collected in 60 minutes before the call to this endpoint. Because each station could have distinct reporting periods (like it will send WX frame every 3, 4, 5 or 10 minutes) the summary is generated from the dataset of different size.

URI: http://pogoda.cc:8080/meteo_backend/summary?station=skrzyczne
Method: GET
Parameters: *{station_name}* – a name of the station to query the WS about, the value of ‘name’ field in GET /listOfAllStations response
Example Response:

```
{
  "last_timestamp": 1618387457,
  "number_of_measurements": 15,
  "avg_temperature": -2.481481,
  "temperature_qf": "FULL",
  "qnh": 1018,
  "qnh_qf": "FULL",
  "humidity": 31,
  "humidity_qf": "FULL",
  "direction": 61,
  "average_speed": 3.52,
  "gusts": 5.28,
  "hour_gusts": 6.16,
  "hour_max_average_speed": 3.52,
  "hour_min_average_speed": 1.76,
  "wind_qf": "FULL"
}
```

Fields of the response

Name	Type	Description
last_timestamp	long	UNIX epoch timestamp when last data has been received from stations
number_of_measurements	int	Number of measurements (APRS weather packets) used for summary calculation. Number of packets received in last 60 minutes
avg_temperature	float	External (outdoor) air temperature
temperature_qf	String	Quality factor for temperature measurements (see Quality Factor description)
humidity	int	Humidity (external or internal – depends on the station)
humidity_qf	int	Quality factor for humidity
direction	int	Wind Direction sent by the station in last APRS weather packet
average_speed	float	Average wind speed sent by the station in last APRS weather packet
gusts	float	Maximum wind speed sent by the station in last APRS weather packet
hour_gusts	float	Maximum wind speed reported by the station within last 60 minutes
hour_max_average_speed	float	Maximum average wind speed reported by the station within last 60 minutes
hour_min_average_speed	float	Minimum average wind speed reported by the station within last 60 minutes
wind_qf	String	Quality Factor for wind

GET /lastStationData?station={station_name}&ascendingOrder={true/false}&isLong={true/false}

Returns a list of last 50 or 2000 weather measurement packets send by the stations pointed by {station_name}

URI: http://pogoda.cc:8080/meteo_backend/lastStationData?station=bezmiechowa
Method: GET
Parameters:
{*station_name}* – a name of the station to query the WS about, the value of 'name' field in GET /listOfAllStations reponse
{ascendingOrder} – OPTIONAL parameter used to switch sorting by a timestamp. Default is 'false' which place the newest measurement as the first one in the response. If set to 'true' the oldest is place at the top of response.

{isLong} – OPTIONAL parameter which controls the time length of the response. By default set to ‘false’ what returns last 50 measurement packets. If set to ‘true’ switches to last 2000 packets.

Example Response:


```
  "list_of_station_data": [
    {
      "id": 683781,
      "epoch": 1618415423,
      "datetime": {
        "nano": 0,
        "year": 2021,
        "monthValue": 4,
        "dayOfMonth": 14,
        "hour": 17,
        "minute": 50,
        "second": 23,
        "dayOfWeek": "WEDNESDAY",
        "dayOfYear": 104,
        "month": "APRIL",
        "chronology": {
          "id": "ISO",
          "calendarType": "iso8601"
        }
      },
      "station": "bezmiechowa",
      "temperature": -1.66667,
      "humidity": 0,
      "pressure": 1015,
      "winddir": 345,
      "windspeed": 0,
      "windgusts": 0,
      "tsource": "SR8WXB-1",
      "wsource": "SR8WXB-1",
      "psource": "SR8WXB-1",
      "hsource": "SR8WXB-1",
      "rsource": "SR8WXB-1"
    },(…) ] }
```


Fields of the response

Name	Type	Description
id	long	An integer identifier in database
epoch	long	UNIX epoch timestamp when that data has been received from a station
datetime	LocalDateTime	Serialized version of LocalDateTime (DateTime in MySQL). Returned in server specific timezone. Depreciated DO NOT USE!!
station	String	Station name. The value of field ‘name’ of /listOfAllStations response
temperature	float	External (outdoor) air temperature
humidity	int	Humidity (external or internal – depends on the station)
pressure	int	QNH
winddir	int	Wind direction reported by the station
windspeed	float	Average wind speed reported by the station

windgusts	float	Maximum wind speed reported by the station
tsource	String	Callsign and SSID which is a source of temperature data
wsource	String	Callsign and SSID which is a source of wind data
psource	String	Callsign and SSID which is a source of pressure data
hsource	String	Callsign and SSID which is a source of humidity data
rsource	String	Callsign and SSID which is a source of rain data

GET /stationData?station={station_name}&from={ts_from}&to={ts_to}

This endpoint returns all weather data received between chosen points in time. The endpoint is limited to return not more than one week of measurements. If a caller will provide more than 7 days of time span to return the data for, an error code 416 (Range Not Satisfable) will be returned.
Method: GET
URI: http://pogoda.cc:8080/meteo_backend/stationData?station=opole&from=1618407724&to=1618417724
Parameters:
{station_name} – a name of the station to query the WS about, the value of 'name' field in GET /listOfAllStations response
{ts_from} – UNIX timestamp beginning the time period to return data for.
{ts_to} – UNIX timestamp ending the time period to return data for.

Example Response: The same as for lastStationData

GET /trend?station={station_name}

This endpoint returns a trend of data in last 8 hours. The backend fetch four data sets from the database. 30 minutes wide windows from two, four, six and eight hours before the endpoint is called. These sets are then averaged and returned along will the last data send by the station

Method: GET
URI: http://pogoda.cc:8080/meteo_backend/trend?station=magurka
Parameters:
{station_name} – a name of the station to query the WS about, the value of 'name' field in GET /listOfAllStations response

Example Response

- {


```
"last_timestamp": 1618417915,
"displayed_name": "Magurka Wilkowicka",
```

```
  "current_temperature_qf": "FULL",
  "current_qnh_qf": "FULL",
  "current_humidity_qf": "FULL",
  "current_wind_qf": "FULL",
  "temperature_trend": {
    "current_value": -1.7,
    "two_hours_value": -1.7,
    "four_hours_value": -0.1,
    "six_hours_value": -1.7,
    "eight_hours_value": -2.2
  },
  "humidity_trend": {
    "current_value": 32,
    "two_hours_value": 32,
    "four_hours_value": 33,
    "six_hours_value": 32,
    "eight_hours_value": 31
  },
  "pressure_trend": {
    "current_value": 1018,
    "two_hours_value": 1018,
    "four_hours_value": 1018,
    "six_hours_value": 1018,
    "eight_hours_value": 1019
  },
  "average_wind_speed_trend": {
    "current_value": 2.2,
    "two_hours_value": 3.5,
    "four_hours_value": 2.2,
    "six_hours_value": 2.5,
    "eight_hours_value": 2.5
  },
  "maximum_wind_speed_trend": {
    "current_value": 3.96,
    "two_hours_value": 6.16,
    "four_hours_value": 4.4,
    "six_hours_value": 6.16,
    "eight_hours_value": 5.28
  },
  "wind_direction_trend": {
    "current_value": 59,
    "two_hours_value": 57,
    "four_hours_value": 57,
    "six_hours_value": 63,
    "eight_hours_value": 55
  }
}
```


Quality Factor

Each parameter handled by the Web Service has its own, corresponding Quality Factor. ParaTNC station controller send not only APRS weather packets but also a telemetry which consist statistics information about the radio network and data about sensor status . ParaTNC sends telemetry data each 10 minutes and it also influences how the Web Service interpret that data and calculate Quality Factor. Web

Service knows how to interpret that data if it is provided by a telemetry version number. This functionality is specific for ParaTNC, so although ParaTNC uses standard APRS telemetry format Web Service cannot interpret coming for other stations equipped with different hardware.

- **FULL** – Sensor works OK and data are fully valid.
- **DEGRADED** – There were some problems in last 10 minutes but some data were obtained and could be treated as valid. The nature of a problem depends of the sensor type. This could means some Modbus-RTU / I2C / One-Wire comm timeouts, excessive slew rate, value out of range etc.
- **NOT_AVALIABLE** – A sensor doesn't work for last 10 minutes or the station isn't equipped with it at all
- **NO_DATA** – Station doesn't transmit any weather and telemetry data for more than one hour and the Quality Factor cannot be calculated