Format specification for the SMET Weather Station Meteorological Data Format version 0.95

Mathias Bavay

July 7, 2010

Abstract

The goal of this data format is to ease the exchange of meteorological point measurements by providing both the data and the metadata in an easy to interpret and to manipulate format. Data manipulations should be possible manually using a standard text editor or using some tools (shell scripting, awk, perl, as well as spreadsheets, R, as well as stadard programing languages). The format is versioned, which means that format updates could break backward compatibility if necessary.

1 Requirements

- it should contain both data and metadata
- it should be "self documented" (not regarding the format, but regarding the data, ie we are not talking about schema!)
- it should contain an arbitrary number of meteo parameters, for an arbitrary number of time steps
- it should support any kind of timsteps (ie: no fixed sampling rate)
- it should be easy to parse with common languages and tools (ie: fortran, c, c++, java, awk, perl, bash)
- it should be easy to preview with common tools (ie: no preprocessing needed even if losing a few timesteps/capability). Common tools include R, gnuplot, Excel, Matlab
- it should be possible to format it in a user friendly, eyes-friendly way (read: columns alignment)
- it should be extensible
- it should be useful for a wide range of applications

2 File structure

The file is structured in three sections: a signature line as the very first line, followed by a header, followed by the data. Lines are [CR] or [CR][LF] or [LF] terminated. Comments are supported, starting with a "#" or a ";" character and extending to the end of the line. A comment can follow values on a line. Empty lines can be found anywhere in the file (in the header as well as in the data).

2.1 Signature

The signature line is used to identify the file type, the file format specification version and the format type (ascii or binary). This signature has a fixed format in order to be easy to parse and the parser can reject any file that does not follow this signature or whose specification version does not match. However, the parsers are strongly encouraged to suppport past specifications and to try parsing more recent versions. The signature is formated as follow:

```
{file identifier} {specification version} {file type}
```

with:

- file identifier: fixed string "SMET"
- specification version: decimal number
- file type: string, either "ASCII" or "BINARY"
- all fields separated by one and only one space

Example:

```
SMET 0.95 ASCII
```

2.2 Header section

The header section contains all the metadata. It starts by a section marker, that is the following fixed string, alone on a line, capitalized:

```
[HEADER]
```

Then, the header section purely consists of key-value pairs. Some keys are mandatory while other keys are optional. The general format for a key-value pair is the following:

```
{key} = {value}
```

with:

- key: any string of alphanumeric characters as well as "-_:.", US-ASCII encoded
- value: any string of character, UTF-8 encoded
- the delimiter can be any mixture and any number of spaces and tabs (at least one)

The following keys are mandatory:

- station_id: identifier for the station, any string
- location information (see below)
- nodata: nodata value that is used in the data section, decimal number
- fields: string of spaces/tabs delimited field types.

The location information must be either one of the following keys:

- latitude: decimal latitude, decimal number
- longitude: decimal longitude, decimal number
- altitude: altitude above sea level, in meters, decimal number

OR

- easting: easting coordinate in a cartesian grid, decimal number
- northing: northing coordinate in a cartesian grid, decimal number
- altitude: altitude above sea level, in meters, decimal number
- epsg: epgs coordinate system code, integer number

If both are specified, they must match to within +/-5m the same location. The location information can be written in each data line (adding the necessary columns) in case of a mobile station. In any case, the whole location data set must be either in the header or in the data section, with the exception of the epsg code always being in the header. Conversions between latitude/longitude and cartesian coordinate systems can be made using the Proj4 (or libproj4) software available at http://trac.osgeo.org/proj/[1].

The following are optional:

- station_name: human readable name for the station, any string
- tz: timezone of the measurements, decimal number positive going east. If not provided, utc is assumed
- creation: timestamp of the file's creation date (ISO 8601 Combined date and time formatted)
- source: string describing the origin of the file
- units_offset: a vector of decimal numbers to add to each value of the matching column
- units_multiplier: a vector of decimal numbers to mutiply each value of the matching column (AFTER potential offset addition)
- comment: a free string to write any comment

[1]The conversion string is built similarly as what follow: cs2cs +init=epsg:{epsg_code} +to +proj=latlong +datum=wgs84 +ellps=wgs84

The field tyes are measurement parameter identifier chosen from the following list:

- TA
- RH
- ISWR ...
- timestamp (ISO 8601 Combined date and time formatted)
- julian. If both timestamps and julian are present, they must be within 1 second of each other.

2.3 Data section

The data section contains the data and starts with a section marker, that is the following fixed string, alone on a line, capitalized:

```
[DATA]
```

Each data point occupies one line and the fields are delimited by any mixture and number of spaces and tabs (at least one). All units are MKSA with the possible use of the above mentionned multipliers/offsets. The data is sorted by ascendant temporal order (the use of a timestamp as the first field makes an alphabetic ascending order equivalent to ascending temporal order).

3 Format variations

The file can be gzipped (as a whole). In such a case, only the gzip signature will be visible and the file will either need to be manually gunzipped before reading or read using libgzip (then giving access to the specific WSMDF signature). A binary version of the format is also available, the only difference with the ASCII version being that the data section (excluding the section marker) is coded as 32 bits IEEE754 single precision binary floating-point format. In such a case, no timestamp can be used, instead a julian date can be used (and must be coded as 64 bits IEEE754 double precision). Each data "line" must be terminated by a carriage return (`'\n'`character).

4 Example file

```
SMET 0.9 ASCII
[HEADER]
station_id = test_station
latitude   = 46.5
longitude  = 9.8
altitude   = 1500
nodata = -999
tz     = +01
fields = timestamp TA RH VW ISWR
units_offset = 0 273.15 0 0 0
```

```
units_multiplier  = 1 1 0.01 1 1
[DATA]
2010-06-22T12:00:00   2.0  52  1.2  320.
2010-06-22T13:00:00   3.0  60  2.4  340.
2010-06-22T14:00:00   2.8  56  2.0  330.
```