

# BUFR: A METEOROLOGICAL CODE FOR THE 21ST CENTURY

**Vesa Karhila (vesa@northern-lighthouse.com), Angeles Hernandez-Carrascal**

Northern Lighthouse Ltd, Reading, United Kingdom  
www.northern-lighthouse.com

## Abstract

BUFR is a table-driven code form (TDCF), defined by the World Meteorological Organization as an alternative to traditional alphanumeric codes. Currently, most meteorological observational data are already being produced and transmitted in BUFR, and according to WMO's migration plan, in the near future all meteorological observational data should be coded in BUFR. BUFR has many good characteristics that make it a meteorological code suitable for the 21st century. However, despite these properties, BUFR is still considered by some as a too complex code which requires a too costly time investment.

In this paper, which is to some extent a position paper, we argue that the difficulties that are often perceived as associated to BUFR, can be overcome with good quality BUFR software. BUFR is indeed a sophisticated code, but carefully designed BUFR software should handle the complexity behind the scenes. The root of a large part of the difficulties associated to BUFR is, in our opinion, that some BUFR software packages unnecessarily reflect that complexity on the user interface.

BUFR should be a challenge for the designers of BUFR software, not for its users, in our view. We discuss the characteristics that such software should have, and also present our objected-oriented answer to that challenge, Cipher SoftBUFR, a family of software products build around a library of standard C++ classes.

## 1 INTRODUCTION

BUFR (Binary Universal Form for the Representation of meteorological data) is a standard for the transmission and storage of meteorological observational data. It is a table-driven code form (TDCF), defined by the World Meteorological Organization (WMO) in the late 1980's, as an alternative to traditional alphanumeric codes (TAC). As the use of computers for handling observations became more generalized, the need was felt, in the meteorological community, for a single code more appropriate for computer processing than TACs.

Currently, most meteorological observational data are already being produced and transmitted in BUFR, and according to WMO's migration plan, in the near future all meteorological observational data should be coded in BUFR or, when communication limitations make BUFR unfeasible, in CREX, an alphanumeric TDCF closely related to BUFR. Within the next few years, TDCFs will gradually replace TACs.

The key characteristics of BUFR are:

- BUFR is binary (this is what the first letter of the acronym stands for). This makes it compact and excellent for transmitting and storing huge amounts of meteorological information. However, this implies that humans need suitable software to generate or decode BUFR data.
- BUFR is portable. There are no dependencies on any specific computer word length or byte order.
- It is a Table-Driven Code Form (TDCF). This means that part of what is needed to decode or generate any BUFR message is stored in tables, and therefore does not need to be part of the BUFR-handling software.

- Extendibility, which stems from being a TDCF. When new observation types appear, tables can be extended, and there is no need to modify general BUFR software packages.
- Extendibility and portability make BUFR deserve its second letter: universal.
- BUFR is designed so that quality control information can be embedded if needed.

BUFR is indeed a meteorological code for the 21st century. However, despite its good properties, result of a careful design that took in consideration the needs of the meteorological communities in a computerized world, BUFR is still often considered, particularly by researchers, as a too complex code which requires a too costly time investment.

In this paper, which is to some extent a position paper, we will try to show that most of the difficulties that are often perceived as associated to BUFR, can actually be overcome with good quality BUFR software. BUFR is indeed a complex code, but carefully designed BUFR software should be able to handle the complexity behind the scenes, and free users from involvement in low-level details and therefore allow them to concentrate on their core activities. BUFR should be a challenge for BUFR software designers, not for its users.

## 2 THE BUFR BLUES

There are a number of complaints - the "BUFR blues" - often heard from programmers and scientists who develop user applications involving the decoding or generation of BUFR messages:

- BUFR is too complex.
- BUFR software packages are not user-friendly, and often lack user documentation which is easily accessible (in all the senses).
- Learning to use BUFR decoding libraries requires a time investment which is too costly, and that is taken away from the main activity, e.g. research or application development. Why not use e.g. NetCDF or XML instead?
- Sometimes, for one reason or another, it is not clear what is actually coded in some BUFR messages.

There are a number of explanations for those complaints. Regarding the complexity of BUFR, indeed BUFR is a complex code, but this should not be a problem for users. The root of the problem is, in our opinion, that some BUFR software packages unnecessarily reflect that complexity on the user interface.

We are aware that some software packages are not user-friendly and that they do lack good user documentation. However, this is clearly a problem linked to specific BUFR software packages, and not to BUFR itself.

Regarding the occasional lack of clarity of some BUFR messages, there are several explanations, all reflecting an inappropriate use of BUFR tables. In order to continue with the explanation, we need to mention, for readers not familiar with BUFR, that there are two types of BUFR tables: 1) master tables, i.e. the usual WMO BUFR tables, and 2) local tables, meant for in-house use when there is a need for new descriptors that are not yet included in master tables. Coming back to possible reasons for unclear content of some BUFR messages, these are, in our experience, the usual reasons:

- A BUFR message has been encoded with local tables, but the user has not received those tables. Note that using local tables for encoding and not distributing them is in obvious conflict with the Universal idea of BUFR.
- There is an error in the BUFR master tables used for encoding or in those used for decoding.

- BUFR “tweaking”. The data are encoded in BUFR, not according to the descriptors of the message, but to some rule which is only known by the creator of the “tweak”. An example is when the actual correspondence between the satellite identifier and satellite name is different from the description given in BUFR master tables. While this is an occasional problem, it represents a serious misuse of BUFR, as it leaves the door wide open to misunderstandings and makes archives useless with time. While the “tweaked” data are being produced, perhaps several people in the organisation are aware of the real correspondence (hopefully everybody that needs to know!), but a few years later, perhaps not even the originator of the “tweak” remembers it.

Finally, regarding the time investment needed to get familiar with BUFR, we see it as a consequence of other points. Indeed, if a programmer or scientist, new to BUFR, needs

- to read WMO Manual #306, which is intended for BUFR software library developers, in order to get a working, basic understanding of BUFR,
- to learn to use poorly documented libraries consisting of routines accessible through complex and cryptic interfaces, and
- to struggle to access appropriate BUFR tables,

then it should not come as a surprise that this user gets frustrated and tries to avoid BUFR altogether, if possible, and accept any alternative that offers a way out.

Alternatives such as NetCDF or XML might look appealing. However, they are essentially wrappers, and in order to make them suitable for the task, physical meteorological standards within the general standard would need to be defined. Such a definition would need years of work and would end in a migration process of the same scale than the current migration to BUFR (Martellet, 2007).

In our view, all the difficulties usually associated to BUFR are more related to specific software approaches, and not to BUFR itself, and they can be avoided with well designed BUFR-handling software which includes appropriate user documentation. We think it is possible to enjoy all the good characteristics of BUFR without suffering the “BUFR blues”.

### **3 WHAT SHOULD BE THE CHARACTERISTICS OF GOOD BUFR SOFTWARE?**

Some of the requirements for a software package stem directly from user’s comments, and others from general good software engineering practices:

- BUFR’s internal complexity should be hidden - users should not need to get involved with low-level implementation details.
- It should have a clean, intuitive interface, accompanied by good documentation describing it.
- Good training material should be available to help new users to get started quickly.
- BUFR tables should be in a format that makes them easy to browse by humans, and also easy to maintain. The same set of BUFR tables should be used by the software and by humans, to avoid any “mistake-by-copy” mismatch errors. Unfortunately, BUFR software packages often have a set of tables used by the software and another in a format, e.g. Microsoft Word, meant for humans.
- Portability, to avoid ties to a particular computer system.
- Reliability, robustness, flexibility.
- It should facilitate the development of applications easy to maintain.

## 4 THE CIPHER BUFR FAMILY

As mentioned earlier, we think that a BUFR software package should not be a challenge for its users. The challenge should be for software engineers to design and implement good BUFR software packages. In the late 1990s we decided to take on the challenge and we created Cipher SoftBUFR, which provides the following solutions to the requirements described in the previous section:

1. Object-oriented design and implementation. This makes it possible to define clean and intuitive interfaces. The Application Program Interface (API) is carefully designed and based on high-level BUFR concepts, hiding all the unnecessary internal low-level complexities from users.
2. WMO BUFR tables are written in HTML format, which makes them easily browsable for users. The same set of tables is used by the software, which ignores the HTML tags. Having only one set of tables common for browsing and for the software prevents mismatches. The internal structure is very simple, which makes table maintenance easier, and therefore errors more unlikely.
3. Comprehensive user documentation, also in HTML format.
4. A tutorial, also in HTML format, helps new users to get started quickly and in a gentle way, allowing them to compile and run small example programs while explaining the basic BUFR concepts.
5. Naturally, it can handle local BUFR extensions, in case there is a need for them.
6. Portability. The core of SoftBUFR is a C++ library, written in standard C++, and therefore highly portable. Our development platform is Linux, but Cipher products are also available for Microsoft platforms.
7. Editions 3 and 4 of BUFR (i.e. those currently operational) are supported. Cipher SoftBUFR can also decode old BUFR messages coded according to now obsolete editions 1 and 2.

### 4.1 Products in the Cipher BUFR family

The Cipher BUFR family is formed by a set of members, all based on SoftBUFR C++ library. All include BUFR master tables in HTML format and user documentation, also in HTML format.

- **SoftBUFR** - This is the full SoftBUFR Development Kit, our star product for handling BUFR. Its core is a library of C++ classes for decoding and encoding BUFR, and also includes an HTML-based tutorial. It is available for commercial use, and we believe it has risen to the challenges described in section 3. Cipher SoftBUFR is already being used to generate many of the BUFR messages produced in the world, as it is embedded in Vaisala's upper air sounding systems.
- **SoftBUFR-Decode** - This is a subset of the full SoftBUFR Development Kit, meant for application programs that need to decode BUFR data. It includes a tutorial in HTML format. SoftBUFR-Decode is freely available for evaluation and for non-commercial use. There is a version for commercial use: **SoftBUFR-Decode+**.
- **BUFRtool** - This program was originally written to demonstrate some of the features available in Cipher SoftBUFR library, but it has turned out to be a very popular tool to study, decode and even encode BUFR messages. It does not require any programming skills. BUFRtool is free for non-commercial use.

BUFRtool can be downloaded from Northern Lighthouse web site (<http://www.northern-lighthouse.com/>) which also gives the opportunity to browse WMO BUFR tables.

## 5 TO CONCLUDE

BUFR is a sophisticated meteorological code, result of a careful design that takes into account the needs of the meteorological communities in a world where practically all the observation processing is computer based. We have discussed some issues often associated to BUFR, such as its complexity or the time investment it requires from new users, and we have argued that the perceived downsides of BUFR are not necessarily associated to BUFR itself, but to specific software packages or to inappropriate use.

We have also described the characteristics that appropriate BUFR software should have, from a user perspective, and we have presented our object-oriented solution, Cipher SoftBUFR, a set of software products that has at its core a library of C++ classes. The use of object-oriented methods has allowed the definition of clean, intuitive, high-level user interfaces. We believe Cipher SoftBUFR shows that it is possible for users to benefit from the good properties of BUFR without suffering the "BUFR blues"

## REFERENCES

Martellet, J. (2007). WMO Strategy for Migration to Table Driven Code Forms. RA VI Training Workshop on BUFR and Migration to Table Driven Code Forms, Langen, Germany, 17-20 April 2007.

Martellet, J. (2007). Why and How the Migration to TDCF. RA VI Training Workshop ON BUFR and Migration to Table Driven Code Forms, Langen, Germany, 17-20 April 2007.

Thorpe, W. (1995). Guide to the WMO Code Form FM 94 BUFR. Federal Coordinator for Meteorological Services and Supporting Research. Silverspring, MD, USA.  
URL <http://dss.ucar.edu/docs/formats/bufr/bufr.pdf>.

World Meteorological Organization (2009). Manual on Code, International Codes, Volume I.2, Part B - Binary Codes. WMO-No. 306.