WORLD METEOROLOGICAL ORGANIZATION

IPET-SUP-1/Doc. 12.1
(13.03.2015)

COMMISSION FOR BASIC SYSTEMS
OPEN PROGRAMME AREA GROUP ON INTEGRATED OBSERVING SYSTEMS

INTER-PROGRAMME EXPERT TEAM ON SATELLITE UTILIZATION AND PRODUCTS

ITEM: 12.1

FIRST SESSION

Original: ENGLISH

GENEVA, SWITZERLAND, 16-19 MARCH 2015

# ecCodes decoding/encoding library and tools

*(Submitted by Enrico Fucile)*

**Summary and Purpose of Document**

ECMWF has developed a decoding library for BUFR and GRIB (ecCodes). The principal aim of the library is to provide easy access to BUFR and GRIB data with a semantic layer that enables the user to make use of the data without having knowledge of the underlying format. This will be of help in the development of tools for satellite data users. The library is going to be published under Apache 2.0 license to allow community development.

## ACTION PROPOSED

The session is invited to:

(a) Evaluate the usefulness of ecCodes for satellite data users;

(b) Take action to gather community resources for the development of tools based on ecCodes;

## DISCUSSION

### Introduction

Many satellite data are exchanged and used for NWP purposes in BUFR format. There are few BUFR decoding libraries available for data users and they all require the user to have a considerable knowledge of the BUFR format and its complex mechanisms of coding. This limitation of the software has produced the misconception that BUFR is a format not suitable for use in research and sometimes also in operational environments. ECMWF has developed a decoding library allowing the user to access BUFR data without any specific knowledge of the underlying data format.

**Semantics access to BUFR data**

BUFR is a binary data format based on a well-established governance model providing a strong support for operational processing and long term archiving of data. From the user point of view the strong governance and the coding rules are an inhibiting factor for an effective use of the data.
BUFR is providing a good governance, but there is a semantics gap that has to be covered to allow the user to make sense of the data without having to know too much of a complex coding system.
The natural semantics of BUFR is based on code figures composed of six digits having meanings that are accessible through the consultation of tables which are made public by WMO in its web site (http://www.wmo.int/pages/prog/www/WMOCodes/WMO306_vI2/LatestVERSION/LatestVERSION.html).
The aim of the new ecCodes library developed by ECMWF is to provide a semantics based on plain text strings linked to the BUFR codes and expressing their meaning in a way that is understandable by the user without need of consulting external documents.
A nomenclature has been developed at ECMWF for the BUFR tables and will be made public with the documentation of ecCodes. Here are some examples taken from BUFR table B, class 12 (temperature)

| *Code* | *Meaning* | *Key (nomenclature)* | *Units* |
|---|---|---|---|
| 012062 | EQUIVALENT BLACK BODY TEMPERATURE | **equivalentBlackBodyTemperature** | K |
| 012063 | BRIGHTNESS TEMPERATURE | **brightnessTemperature** | K |
| 012064 | INSTRUMENT TEMPERATURE | **instrumentTemperature** | K |
| 012065 | STANDARD DEVIATION BRIGHTNESS TEMPERATURE | **standardDeviationBrightnessTemperature** | K |
| 012066 | ANTENNA TEMPERATURE | **antennaTemperature** | K |
| 012070 | WARM LOAD TEMPERATURE | **warmLoadTemperature** | K |
| 012071 | COLDEST CLUSTER TEMPERATURE | **coldestClusterTemperature** | K |
| 012072 | RADIANCE | **radiance** | W m-2 sr-1 |

Table 1

In most of the decoding software, including the current operational ECMWF decoder BUFRDC (https://software.ecmwf.int/wiki/display/BUFR/BUFRDC+Home), the user has to access the item of information through the code. As an example from table 1, to retrieve the brightness temperature the user has to deal with the code 012063 and should check the version of the table as in some cases the meaning can change from one version to another. This last check is never done and is source of confusion.
ecCodes provides a codes_get function which is returning the array of values of brightness temperature making a request through the key name in a syntax like

```
x=codes_get(message,"brightnessTemperature")
```

where x will be an array containing the values of the brightness temperature retrieved from the message. This provides a high abstraction layer for the user who can access the BUFR message without any need of knowing the BUFR codes. The library provides a further help in accessing data and their meaning in the message as the units are easily retrieved as an attribute of the key name with the syntax

```
u=codes_get(message,"brightnessTemperature->units")
```

where u will be a string variable having the value "K" for Kelvin.
The maintenance of the nomenclature is of fundamental importance in this context and ECMWF has developed a database of BUFR tables with key names for all the table versions including some local EMCWF tables. The database is complemented by tools to build the required configuration files for ecCodes and a web application is being developed to expose the content to the users.

### Accessing data structures

BUFR messages and especially satellite data are not made by flat data vectors. They are stored in the message as tree data structures. There is therefore the need to provide a conditional access to the data elements. ecCodes provides two different mechanisms: access by rank and by condition.

**Access by rank**. The array retrieved is the nth occurrence of the named variable in the tree. An example is the second “backscatter” (for a scatterometer) or the fifth “scaledIasiRadiance” for IASI data. In the ecCodes keys language the two examples would be

```
x=codes_get(message,”backscatter#2”)
x=codes_get(message,”scaledIasiRadiance#5”)
```

The rank is expressed after a hash sign “#” in the key name.

**Access by condition**. A condition on the tree branch to be selected is given for the retrieval of the data array. This would allow to select the “backscatter” with beam identifier equal 2 for scatterometer data or “scaledIasiRadiance” for channel 1145. In the ecCodes keys language the two examples are

```
x=codes_get(message,”/beamIdentifier=2/backscatter”)
x=codes_get(message,”/channelNumber=1145/scaledIasiRadiance”)
```

The condition is expressed in a directory like syntax very natural to remember.

### Conversion to json and xml

Web based technologies are relying on two data formats for which a wide set of tools is available: json and xml. At ECMWF json has been chosen for the web development and ecCodes is providing a command line conversion tool (bufr_dump) producing json from a BUFR file. The json format is quite easy to learn and provides a readable layout. Here follows a small part of a json dump for an IASI dataset:


```
[

  {
    "key" : "channelNumber",
    "value" : 1145,
    "index" : 2352,
    "code" : "005042",
    "units" : "Numeric",
    "scale" : 0,
    "reference" : 0,
    "width" : 14
  },
  {
    "key" : "scaledIasiRadiance",
    "value" :
    [
      2536, 2848, 2703, 2691, 3020, 2730, 2721, 3002, 2777, 2887,
      2801, 2897, 2655, 2898, 2831
    ],
    "index" : 2353,
    "code" : "014046",
    "units" : "W m-2 sr-1 m",
    "scale" : 0,
    "reference" : -5000,
```

```
    "width" : 16
  }
 ]
```

In json format any object is delimited by curly braces and can have several attributes. Two objects are present in the example with an attribute “key” which defines the name of the element as “channelNumber” and “scaledIasiRadiance”.
A similar conversion to xml format will be implemented to provide the user the opportunity to use the many xml tools available.
The conversion to these two popular data formats is a key characteristics of the library and opens the data to the use of tools which are already used in several other data processing contexts.

**Languages and tools**

ecCodes is designed to provide most of the data access features through the simple key name syntax described in the previous paragraphs. The same approach is valid for all the languages for which at the moment ecCodes provides interfaces which are: C, Fortran and python.
The extension to other languages is possible and fairly easy to implement as only few functions are provided by the library and all the data access features are delegated to the keys syntax.

**Collaborative development**

ecCodes will be released under Apache 2.0 license which allows a large freedom of use and modification of the code base. ECMWF has also developed guidelines for a collaborative development of the library and is willing to accept contribution to be integrated in the main branch.
The realisation of a community project under the WMO umbrella is desirable and would be of great help for the satellite community to develop tools for data processing and visualisation.
The areas for which a collaborative development is foreseen to be more effectively applicable are:

- development of new language interfaces like IDL, R, matlab, perl, …,
- development of data template for satellite data
- graphical tools
- processing tools
- examples for the users

Contribution from developers with specific knowledge on these areas would be very welcome and would bring clear benefits to the satellites users community.