User-friendly presentation of BUFR data

Serge Savchenko, Empress Software Inc., Canada

Abstract

The transition to Table Driven Code Forms (BUFR and CREX) as a means of meteorological observational data exchange is inevitable. This process does not need to be an affliction; in fact, it is relatively easy to assume a pro-active role with the right tools at your disposal. Empress database management system is an all-in-one solution for managing observational data packed in Binary Universal Form for the Representation of meteorological data (BUFR.)

Empress is capable of ingesting BUFR data in real time, as well as decoding and analyzing using Persistent Stored Modules. This paper illuminates the framework for managing BUFR data using the Empress database and for presenting decoded data by means of a user-friendly interface.

Background

WMO standardized on BUFR in 1988. The standard allows for a transparent exchange of observational data between multiple meteorological entities. The key to the transparency is the inclusion of BUFR data descriptors within BUFR messages. Thus, a party unaware of the existence of atypical observational data can automatically decode BUFR messages. BUFR is a table driven code. It is infinitely extensible. However, the acceptance for BUFR is rather slow, mainly because humans cannot read BUFR without first being decoded. While there are a few BUFR decoders available, many are setup as large batch jobs and lack user interactivity. In other words, they require practical knowledge and large computer systems.

Objective

Easier management of coded messages would promote wider acceptance of the table driven codes, such as BUFR and CREX.

Empress Software has produced a software system (Browse-BUFR) that allows users to interactively browse through BUFR messages. The system was created as a proof of concept with the objective to allow the viewing of observational data stored in BUFR messages via popular software products such as browsers and MS-Excel.

Specifications

The specifications for Browse-BUFR are as follows:

- meteorological data is stored in BUFR
- data be decoded selectively 'on-a-fly'
- data be extracted interactively according to user requirements
- data be presented to a wide range of end-user products

Publishing BUFR Data on the Internet

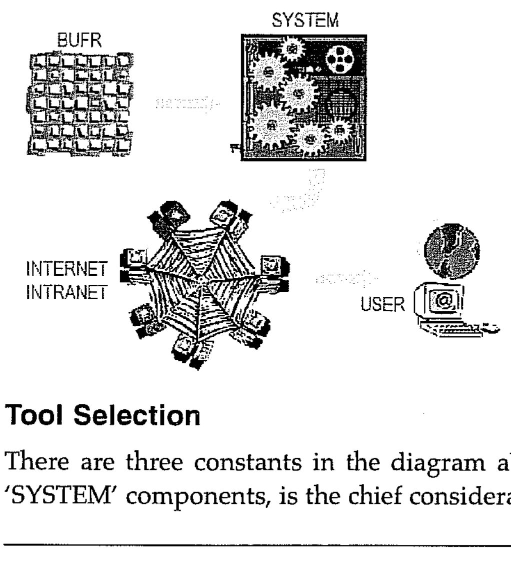

Figure 1

Tool Selection

There are three constants in the diagram above (Fig. 1.) The only variable is the entity called 'SYSTEM.' The 'SYSTEM' components, is the chief consideration of the Browse-BUFR implementation.

The two main components comprising Browser-BUFR are Empress RDBMS and Tomcat JSP Server.

Tomcat JSP Server is a platform independent Internet server that provides a logical connection between the Internet browsers of remote users and the local database server containing BUFR records.

The Empress database management system was chosen as the 'holder' for BUFR messages for the following reasons:

- Multi-platform capability, i.e. one application can be deployed on Windows, Linux and UNIX operating systems
- Wide range of Application Programming Interfaces (APIs): ODBC, JDBC, HTML, Perl, PHP, and Python
- Capacity to ingest BUFR data at high rates and store it as Binary Large Objects (BLObs) in a database
- Ability to embody BUFR decoders into Empress User-Defined Functions and store them as meta data inside a database; to wit, uniting the self-described code (BUFR messages) with applicable executable logic in a single logical entity such as a database
- Extending this functionality to all high-level APIs, such as SQL, ODBC, JDBC, HTML, Perl, PHP and Python

Methodology

Three steps are required to complete the Browse-BUFR system:

1. Insertion of BUFR observational data into an Empress database
2. Embodiment of BUFR decoders into Empress User-Defined Functions
3. Development of an interactive end user tool to allow search and query operations on BUFR data stored in an Empress database

BUFR Data And Meta Data

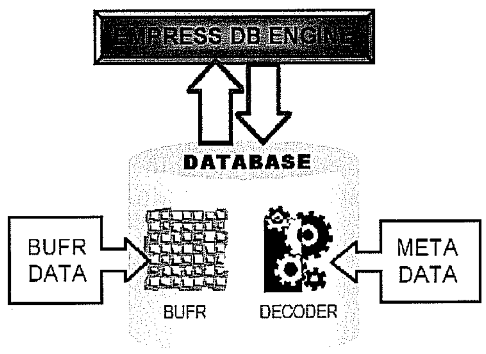

Figure 2

Methodology – Step 1 – BUFR Messages Ingest

The process of ingesting BUFR data into a relational database is hampered by the non-relational nature of the table driven code. The task requires BUFR data to be first decoded and then mapped into database table attributes. While it is possible to accomplish and is at the present time the preferred way of preparing observational data for analysis, this process is involved very costly computer time. Another major drawback is the amount of storage space decoded BUFR data occupies. For this reason, it is not feasible to store decoded BUFR data in relational format for a long term.

However, it is effective to store coded BUFR messages in a relational database for a long term. An Empress database system is capable of ingesting BUFR data at the rate of over 1000 messages per second. In practical terms, a six-hour BUFR sample provided by ECMWF was inserted into Empress database in 166 seconds on a Celeron 1.70 GHz system with 512 MB of RAM. In other words, BUFR data can be ingested directly from a live feed or from an electronic file.

The recommended method for ingesting BUFR messages into a database is partial decoding. BUFR messages consist of sections. Sections 1 and 2 are of fixed length and can be easily and quickly mapped into database table attributes. When implemented, partial decoding instantly creates a classification for each BUFR message. This enables a user to quickly pinpoint any message based on date, time, originating centre, etc.

Methodology – Step 2 – Embedding BUFR Decoder Into Database

The larger coded portion of a BUFR message is stored as a Binary Large Object (BLOb) data type of an Empress database. It allows for the most effective use of storage, but remains indecipherable without a BUFR decoder. Today external BUFR decoders 'pull' observational data out of a database, decode it and store the data elsewhere for analysis. This process works well for large sets, however the decoded data is not presented in a 'user-friendly' fashion and is not kept after the analysis is complete for it takes too much space. This implementation makes it impossible for an individual to interactively work with BUFR data.

A different approach is needed for 'personalizing' the management of BUFR data.

If BUFR data is kept in the database in its original form, then every time the data is 'pulled out' from the database, it needs to be decoded and presented in a 'user-friendly' format. To achieve this, the decoding logic must to be embedded into the database itself and kept on the 'inside' of the database. In other words, it becomes metadata in the database. Thus both BUFR messages and its decoding logic make up a single logical entity to any tool on the 'outside' of the database.

The only requirement for an existing BUFR decoder to become metadata in the Empress database is that it is implemented in 'C' programming language. Then it can be included into the Empress database as a structure known as a Persistent Stored Module. Empress Persistent Stored Module consists of any number of user-defined functions (UDF), user-defined procedures (UDP), and user-defined operators (UDO.) Once the BUFR decoder is 'wrapped' into a user-defined function, it will be recognized by all high level Application Programming Interfaces (APIs), such as SQL, ODBC, JDBC or HTML/XML. This enables users of applications like MS-Excel, JAVA based applications, and Internet Browsers to query BUFR data stored in the Empress database and bring deciphered BUFR messages into personal software of their choice.

Methodology – Step 3 – Completing Browse-BUFR System

Once BUFR messages are stored in the Empress Database as BLOb attributes and the BUFR decoder is 'wrapped' as user-defined functions, a mechanism for performing interactive queries into the database is the only remaining piece of Browse-BUFR system. In this proof of concept, Java Servlets were chosen for a browser based interactive menu. Tomcat JSP server was installed to provide Internet access to Browse-BUFR system. This enabled BUFR message viewing by any user with a browser and connection to the Internet. In parallel to JDBC access via Tomcat, ODBC access to the database was added for Microsoft Excel users. While the latter took seconds to implement, it enabled BUFR message upload into Excel spreadsheets, and subsequently, independent analysis of the meteorological data.

Benefits of Including BUFR Decoders as Meta Data in Database

The main benefit of setting up a BUFR decoder as a user-defined function is the ability to present BUFR messages in a 'readable form.' The de-coupling of BUFR decoders and interfaces renders the 'readable form' to be independent of a user's computer environment. More specifically, suppliers of BUFR messages do not need to concern themselves with the hardware and software requirements of BUFR recipients.

Additional benefits include:

- Qualifying searches on sets of BUFR messages
- Ability to browse through elements of a single BUFR message
- Improved management of BUFR reference tables
- Unification of BUFR messages and its decoders in a single logical entity
- Simplified exchange of BUFR messages between various organizations

Summary

The complexity of Binary Universal Forms for Representation of meteorological data presents considerable difficulty in adoption of the standard. The unification of BUFR messages and its decoding logic into a single database entity allows for easy access to BUFR messages. The user-friendly access creates a high degree of manoeuvrability in management and analysis of observational data stored as BUFR. The user-friendly presentation of BUFR data will promote its acceptance and widen its usage.

References

Martellet, Joël WMO strategy for migration to table driven code forms, ECMWF workshop proceedings, Eighth Workshop on Meteorological Operational Systems, 12–16 November 2001

Bergès, Jean Claude Support of WMO Binary Format (BUFR and GRIB), Proceedings of the Open source GIS – GRASS user conference 2002, 11–13 September 2002