World Meteorological Organisation Spatio-Temporal Data Standards

Similarity to ISO 19100 Geographic Information Standards

Executive Summary:

This paper describes WMO's BUFR (Binary Universal Form for Representation) code form both in text and graphically in UML, with a commentary on BUFR and ISO concepts.

BUFR is a Domain Specific Language. It is a data-driven code form: part schema language; part coding language; part data exchange format; part information model, but it is very large and highly specified. It is viewed as very complex, but the apparent complexity is really considerable detail. The domain of BUFR encompasses all topics of meteorology from observing the weather from land, air, space, on and under the sea, to agricultural meteorology: from types of precipitation to reports of locust swarms.

WMO's BUFR Tables contain all the vocabulary, reference, and code specifications for BUFR. There are ~450 BUFR Tables with ~12000 table records.

Expressed as a potential ISO Feature Catalogue, there are 300 feature collections and 800 simple features. However there are also 1500 feature modifiers, which allow the context and meaning of simple features and collections to be modified, making new features.

Why should we want to decode BUFR into an ISO-compliant GML application form?

BUFR is for specialists. Some of the information in BUFR messages needs to be extracted and made available to non-specialists in a form that commonly available software is able to use, in language the non-specialist can understand. An ISO-compliant, GML application is one way that can happen.

The paper describes a route map to decode BUFR in XML. It outlines a process to present common BUFR features as a feature catalogue of manageable size within a GML application schema.

Gil Ross Met Office

Table of Contents

1. Historical introduction:

WMO, as the specialist UN Agency for Meteorology, have been sponsoring data standards since the UN passed a Directive in 1963 which set up the World Weather Watch.

Initially the telecommunications methods were teletype and radio-transmission via Morse code. So all the initial data codes were human readable (in that they were coded, transmitted and decoded by humans) but they were highly formalised and based on hand operated Morse standards, mainly organised in 5 groups.

These Traditional Alphanumeric Codes (TACs) are still in full use. Hundreds of thousands of messages with these codes are received every day from every corner of the globe. There are around 50 such code forms defined in the WMO Volume 306, Manual of Codes Part A, and these are still assiduously maintained by the WMO Expert Team on Data Representation and Codes.

The replacements for the TAC forms are also in wide use. These are commonly called "Table Driven Code Forms", because both the elements and the non-numeric values which the codes contain are predefined in many tables (at least 450 tables for those maintained by ET DR&C). There is a program to migrate individual TAC forms to table driven codes, but there are still valid reasons for using the TAC forms – they can be written and read by humans. Very many people, particularly in aviation, still have the skills to read these codes. There are possibly 2-3 million pilots, no matter what language they use, who read the METAR and TAF codes by which weather conditions and forecasts for airports are sent to en-route aircraft every ½ hour.

In 1989, FM92 GRIB (GRIdded Binary) was accepted as a new format for gridded 2 dimensional scalar data, and is now extended to multi-dimensional tensor data. This has almost entirely replaced the original TAC gridded codes because of enormously superior data compression capabilities.

At the same time the binary code FM94 BUFR (Binary Universal Form for Representation) was introduced for discrete (and mostly observational) data, and the non-binary version FM95 CREX (Character form for the Representation and EXchange of data) was approved in 1994, but only went operational in 2000.

This paper describes the model of WMO BUFR, and explains commonalities and differences with the ISO 19100 standards for Geographic Information Standards. Since CREX is essentially a character representational form of BUFR, and so it is implicitly included. GRIB however is not considered here because, while many of the concepts explored apply to GRIB too, GRIB is also a simpler model than BUFR. It is only where it develops concepts of multidimensional grids, that GRIB extends the modelling framework.

1.1 BUFR in the scheme of things…

BUFR is a Domain Specific Model and Language[1]. Its specific application is Meteorology, but its domain is any data type which can be fully described in a formal table structure. While WMO maintains the basic BUFR table set, there are many "local" tables and additions to tables, and in the oceanographic domain there is an entirely separate set.

BUFR is highly detailed rather than complex (though 20 years of development have added niche complexities). Some of these additional model structures are ignored in this presentation, but the basic BUFR model maps well to the ISO 191xx series of Geographic model standards. Indeed, the dynamic specification aspects of BUFR are not available in the ISO paradigm.

BUFR is a data-driven code form: part schema language; part coding language; part data exchange format; part information model, but very large and highly specified. It is dynamic in essence. As ISO 191xx prescribes essentially static feature catalogues, so only static (or predefined) portions of the BUFR model are transformable under an ISO model.

This paper describes BUFR in modern modelling terms in a UML description. It describes a roadmap in transforming BUFR messages or instances to an XML variant, and to transform the BUFR model to use the ISO 191xx models and the OGC Observation and Measurement prescription for transforming to GML (Geographic Markup Language).

This process is not intended to convert all BUFR to a GML application: BUFR is too large.

Instead the process is to allow subsets of data which are coded, exchanged and archived in specialist BUFR to be extracted and aggregated to satisfy a non-specialist's query. It is a process which allows a non-specialist to ask for "all temperatures for South West England at a specific date and time" to be mapped to the BUFR elements; extracted from each BUFR message; aggregated and expressed in a GML grammar and delivered to the recipient. This can then be used in commonly available software, translated from the specialist vocabulary into language the non-specialist can understand.

1.2 Summary of the paper

The audience for this paper is likely to be both WMO and ISO/OGC experts and users. This means that the paper is a sort of juggling act between different experiences, attitudes and terminologies. What this document is not, is a coding manual for BUFR.

Section 2 is a review of BUFR as it might have been seen by its designers, and how they managed successfully to reconcile conflicting requirements.

[1] http://en.wikipedia.org/wiki/Domain-specific_modelling

Section 3 tries to unravel the BUFR concepts and explain them in more familiar terms to software and standards developers.

Section 4 is a description of BUFR modelling (and to a lesser extent, coding) in UML and explanatory text. While one picture is worth a thousand words, a UML class diagram alone can be even more confusing.

Section 5 describes a roadmap in transforming BUFR messages or instances to an XML variant, and to transform the BUFR model to use the ISO 191xx models and the OGC Observation and Measurement prescription for transforming to GML (Geographic Markup Language).

In section 6 are the conclusions and recommendations.

Section 7 contains the references to WMO BUFR papers and tables, and section 8 is a glossary.

The concluding annex has a short description of the components of a UML class diagram.

2. BUFR – as seen by the designers.

Although the TAC codes are tightly defined and over the years have been rigorously maintained, they require the use of specific decoders in any automated system. The decoders seldom were designed for separation of the business rules and the decoding application. So the information in the TAC forms is complicated to modify, if only because there are many decoders which need to be modified too.

Although the BUFR design wasn't quite described in the following terms, the expert team had a number of design requirements.

2.1 BUFR Design Requirements

BUFR was to be a general purpose system for exchanging data. To do that, the information content and the coding forms including the (de)coding software were to be separated.

- This meant that BUFR was designed to be much more flexible and expandable than TAC forms.

- BUFR information was to be fully self-describing. By this, it was meant that both the form and content of the information in a BUFR message was to be contained in the message itself. (BUFR was being developed at the same era as SGML, and the designers were aware of the work[2])

- BUFR was to be extremely well compressed. This is the antithesis to the SGML offspring XML, which has a design constraint that:
  "Terseness in XML Markup is of minimal importance."

- An important design goal was to make the "obvious" explicit. This meant exclude as much "self evident" information as possible, instead requiring that "meaning" be specified by definite rules.

The way in which these contradictory requirements were achieved, and the enormous scale, completeness and authority of the BUFR modelling, is impressive indeed.

The result however has never been clearly described and much of BUFR documentation is recondite. Although the BUFR documentation is clearly published and the BUFR tables equally so (see the references in chapter 6), the documentation is unyielding. While there is software freely available, it is aimed at the specialist. BUFR has a poor reputation outside of operational meteorology.

In part the documentation is complicated because the BUFR model is complex, but it also reflects the practices of the era in which BUFR was designed. In 1988, the emphasis was on how to code and decode BUFR – NOT on the modelling process, and it is on BUFR decoding that most of the documentation focuses.

[2] The designers took the concepts of context-free grammars, but did not adopt the hierarchical tree structures and accepted a denormalised solution for efficiency. However, the "coordinate" operators (see also 4.4) and the Table D sequences can be seen as BUFR's solution to defining tree structures.

2.2 How Compression and Self-Description work together.

Rather than describing in detail the coding/decoding process, here we shall discuss the principles behind the algorithms.

- The two, seemingly contradictory, concepts of compression and the self-description in BUFR are gained by making everything, tags and values (except number values), a reference. The references are indexes to elements[3] (rows) of published tables. Indeed a considerable amount of effort was put into condensing the volume of the tags, using all of the techniques below.

- The tags – element descriptors – and the values are grouped apart. When expanded, all the tags are in sequence with all the values, but the tags are grouped separately from the values.

- Number values are compressed by closely defining the precision, scaling by an exponent and removing any reference value for the mantissa for each element in the tables. This makes every number an unsigned integer stored in a predefined bit length binary.
  While this is defined at the application level, it introduces what is essentially a presentational decision into both the coding and the data model.

- As well as each number-value element having predefined compression, there are dynamic redefinition operators, by which temporary precisions can be defined.

- Some common elements which could be subject to varying precisions are duplicated as a different element, but are really only the same parameter with a different precision.

- Further compression is gained by explicit grouping of terms in predefined collections of elements in a template reference. This is made modular by allowing collections to refer to other collections of templates.

- There are dynamic replication mechanisms, which in conjunction with the templates or collections can allow a variable replication of element sequences. Similarly there are dynamic repetition mechanisms to allow repeated values to be compressed in effectively a run-length encoding.

- Yet further compression and enormous flexibility can be achieved by developing a dynamic grouping mechanism which BUFR calls a generalised "coordinate" mechanism. In application this is a single "grouping" mechanism to assign properties to a group of features. In ISO terms, this generalised coordinate can be considered at the same time as a feature collection

[3] Many of the terms used here have specific meaning, and are likely to vary across different modelling themes. Database modelling, UML modelling, XML modelling, different programming paradigms all use terms such as element, row, attribute, descriptors, tables even, arrays, lists etc. with different meanings. Contrariwise, identical concepts are often assigned different names in different canons.

mechanism, a coverage grid mechanism, and a feature attribute mechanism. These generalised coordinates can be horizontal position, or they can be identifiers, declarations of common instrumentation, temporal or (generalised) vertical coordinates or they can be "significance" qualifiers[4].

- There are also packing mechanisms in which array structures can have dynamic offsets removed to reduce the bit length of individual members.

2.3 Later additions to BUFR mechanisms

There are other mechanisms which have been added over the years for different tasks, which have complicated further the BUFR mechanisms. As with many standards, the added functionality makes the maintenance much more difficult, and leads to confusion, not clarification.

However, it seems than many of the added mechanisms are niche requirements. Although it isn't declared so in the documentation, from sampling the actual practical implementation in BUFR messages, few of the added mechanisms were intended to work together. Where complications are used, they seem to be used in isolation.

This means that some of the complications added in later years can be ignored here as they are coding/decoding complexities. Even if they affect the data model, they do not impact the basic structure and can be disregarded from the present modelling analysis.

As an example, in this report the data quality mechanisms are not considered. This is a specialised data modelling process which was added to the original BUFR design, but it also mixes the modelling and coding functions of the BUFR tables which complicates the already complex BUFR model for WMO data exchange. Table C, which otherwise has only coding operations, has had data modelling complications added. This should best be treated separately in follow-on work as a sub-model of the BUFR information model[5].

[4] While the grouping mechanism is undoubtedly efficient in space, because it functions as 3 different XML/GML mechanisms, it also brings some, possibly arbitrary, choices in re-creating the underlying tree model. It might be seen as a "locally efficient" serialisation.

[5] Section 3.1.6.7 of the FM94 BUFR Layer 3 detailed description also recognises that the quality assessment mechanisms are a specialised addition to BUFR, and assigns the explanation to an appendix.

3. The WMO BUFR standard.

BUFR was designed for any data type which could be fully described in a formal table structure. It is primarily a code table or fixed format numeric type, where the values are individual. Although it can also code text strings and arrays, it is not efficient for these. For multi-dimensional array data, the WMO GRIB format is considerably better.

However, the mechanism for table creation allows the message to specify what tables are used, who published them and in what version. The BUFR model also allows the BUFR tables to be self described, and BUFR messages can be used to exchange new versions of the BUFR codes, themselves coded in BUFR.

As well as the ability to specify the whole table structure, there are also mechanisms to allow local definitions within the specification, or to modify many of the global specifications temporarily.

3.1 BUFR as a data-driven form

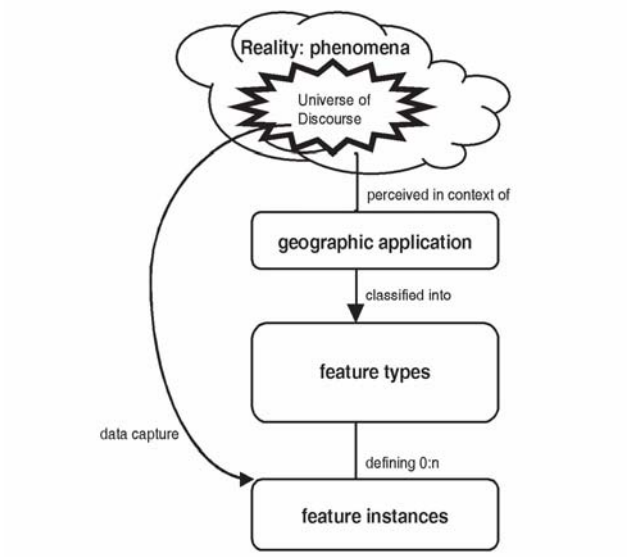
The BUFR model is self-describing. For BUFR, this means rather more than just containing a description and reference to everything in a BUFR message or instance.

BUFR is a data-driven code form. The BUFR model in part is a schema language; in part it is also a coding language. It defines the information model, but it also defines the data exchange format. The data exchange format makes each BUFR instance contain its own schema, like a DTD in XML. More than that, the format allows new features to be defined in a BUFR instance, because there are operators both for coding the message, but also for allowing modification of the predefined features in the BUFR tables. This means that listing all possible features with all possible feature attributes in not feasible: there are too many[6].

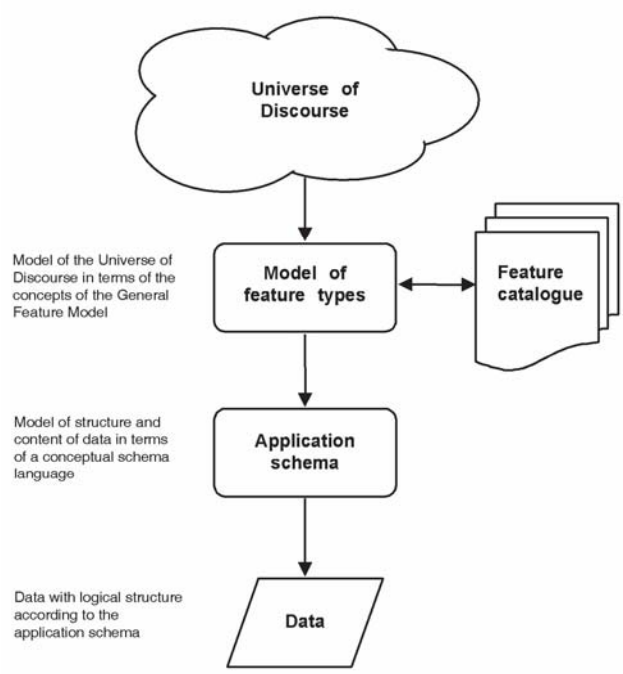
The ISO 191xx models are designed for more static feature catalogues. In ISO 19109 (Rules for Application Schemas) Figure 3 (shown below) describes the process from universe of discourse to data. Here the feature types are separated from feature instances. BUFR allows feature types to be modified within feature instances. In ISO 19109 Figure 4 (below), describing the process from reality to geographic data, the Application Schema follows the Modelling of Feature types, which is intended first to create and then use the Feature Catalogue. BUFR instances hold their own Application Schema, which, again can modify feature types.

BUFR also has model operators which perform three types of operation in GML and ISO: grouping operators, coverage grid operators and feature attribute operators. Since the single BUFR operator type performs the work of 3 in ISO/GML, it is likely that in expressing BUFR in ISO/GML, it will be difficult to decide which ISO function is the most appropriate description from the BUFR model definitions alone.

[6] With 450 Tables and 12000 records expressed as 300 feature collections, 800 simple features and 1500 feature modifiers which can be used in sequence, there are many millions of possible feature types definable.



ISO 19109 Figure 3 - The process from Universe of Discourse to data



ISO 19109 Figure 4 - From reality to geographic data

ISO 19110 defines Operations under GF_Operation. However, it seems that these operations change only the ***value*** of a feature property or attribute. BUFR model operators change the ***meaning*** or ***context*** of a feature or add new attributes a feature, making a new feature. So the BUFR Operators and General Feature Model operations are different things

It is possible that the "AbstractFeatureTypes" in GML may be used to create a "ModifiableFeatureType" and to create "FeatureModifiers" as a general "FeatureAttribute" modification, but this has to be decided.

So only static (or predefined) portions of the BUFR model are capable of being transformed to reflect an ISO model.

Section 5 describes a route by which BUFR messages/instances can be converted to XML. It also describes a process by which the feature types which are actually used in any BUFR Message to be accumulated in a Feature Catalogue. This is a process by which it can be done. It may not be realistic to do so, since such a Feature Catalogue will be very large, and anyway, the basic model is the BUFR Model through specification of the BUFR tables.

What it does allow is for different datatypes or subsets of general data types to be created in a consistent manner as a Feature Catalogue. With this, external users can have access to commonly used types or subsets of BUFR data in a GML form.

This should avoid the immense problem of casting all BUFR data in GML. It isn't necessary.

3.2 Operators, coding operators and model modifiers

There are two types of operators which control these self-modification mechanisms, coding operators and model definition operators. It was originally intended that the

coding operators and the model definitions and their operators were to be cleanly separated in different tables, but over the development and operational stages, the result is not so clear-cut, and there are coding and model definition operators in the main Tables B and C.

The model definition operators make the BUFR information model, in part, a data-driven model. Although the data specification may only be a reference to a pre-defined Table D specification, the BUFR model definition with operators allows a completely flexible data specification. Thus BUFR operations can define new and previously unspecified feature types with new feature attributes.

The main WMO table sets are published on the web (see references in chapter 6), as are many locally defined tables and extensions, but given the period when BUFR was designed, the publications are static documents, intended to be hand extracted and implemented into bespoke software by specialists.

3.3 BUFR Tables, references and indexes

BUFR tables are the response to many requirements. The tables are organised for separate purposes initially, but some of the model functions and some coding functions are spread between the table types.

3.3.1 BUFR Table types

Firstly there are Common Coding Tables, where definitions used across different code forms (specifically BUFR and GRIB) are defined. These tables are external to both the model and coding functions, and for example, list all the data creation authorities across WMO[7].

Table A has the function of defining the sub-catalogue type as part of the full BUFR feature catalogue. The 20 sub-catalogues represent different high-level use cases, but this level of categorisation is only partially followed in the collections of features and classes of simple features, because of re-use of the lower level classes.

Table B contains the list of simple features. These are grouped into 30 classes at present, and there are around 1200 simple features in these classes. At a higher level, the first 9 classes hold the BUFR coordinates, which is a grouping property and by which the BUFR process akin to ISO coverage grids are defined. Table B is used at both the modelling and coding levels.

Table C contains coding operators. Table C also allows some dynamic feature properties, such as annotation and associated features, but mainly it modifies the precision attributes of existing, more static Table B features[8].

[7] The IOC (Intergovernmental Oceanographic Commission) and JCOMM (Joint IOC-WMO technical Commission on Oceanography and Marine Meteorology) had a precursor coding form to BUFR called IOC General Format 3 (GF-3). They now have their own complete set of BUFR tables, and many organisations have their own expanded tables. BUFR has mechanisms which were designed to allow both methods of table expansion.

[8] As explained in section 2.3, this is not true if the Quality Assessment modelling is considered as this introduces model modification functions into Table C. In this paper Quality Assessment is left out.

Table D defines collections of Table B features, and being modular, collections of other Table D collections. Table D is also the templating table, by which exchange formats are predefined.

3.3.2 BUFR Indexes

Tables B, C and D and a single row "table", the replication operator are coded up with an indexing algorithm. All tags in a BUFR message or instance (rows in the tables and code tables, described below) are referenced by this indexing algorithm. The index has three parts (replace Z with B, C or D)[9]:

F: Table_**Z**_indicator: is descriptor (row, tag or element) type. This takes the value 0 for Table B references, 1 for the replicator operator, 2 for the coding operators in Table C, and 3 for the "sequence" descriptors, the collections or templates of Table D.
F occupies 2 bits and can have 4 values

X: Table_**Z**_elementClass: is the sub-class of Table B or D tables, an operator for Table C, and a "number of terms" for the replication operator. This is a type of "namespace".
X occupies 6 bits and can have 64 values.

Y: Table_**Z**_codeValue: references the individual descriptor, feature or feature collection in Tables B and D, and the operator in Table C. For the replication operator it is the number of replications to be performed.
Y occupies 8 bits and can have 256 values.

In Table B, features are assigned base units and precisions. The units are all in SI units, e.g. all in Kelvin for temperature[10]. For features or coverages of discrete variables, the unit may refer to an enumeration – called a code or flag table. There are 370 code and flag tables, and all are referenced by the same index that their table B feature has.

A code table in WMO terms is an exclusive enumeration which may or may not be complete (e.g. not complete – means that there may be reserved code values, exclusive here means that only one value is valid at the same time). A flag table is usually complete, but is not exclusive, i.e. multiple flag values are allowed. Flag values are coded as powers of 2, so that a bit pattern can hold simultaneous flags.

[9] The GF-3 data representational form had tables where all features could be considered either as a feature or a "coordinate" or coverage operator. This choice of splitting 2 bytes into an index was a pragmatic decision which balanced sub-table sizes and table classes (which function as namespaces). An alternative choice would have been to have another bit, taken from X or Y, to designate whether the tag identified a descriptor or a coordinate operator. Many Table B features could, in principle, operate as either. The existing choice forces some coordinate operators (Table B classes 0 to 9) to be replicated as non-coordinate classes. Specifically, Table B classes 26, 27 and 28 replicate as non-coordinate locations, the coordinate operators of classes of 4 (time) and 5 and 6 (1st and 2nd horizontal coordinates), and the 3rd or vertical coordinate class 7 is replicated as the vertical features of class 10.

[10] Although an exception has been agreed for non-SI aviation units, such as "kilo feet".

3.3.3 BUFR references within a BUFR instance

A BUFR message or instance references a code and version number for which BUFR tables are being used – the WMO current tables or local tables. From then on all the collections, elements or operators are referenced by index F-X-Y coded in 16 bits.

To decode the BUFR instance, the references are followed through, e.g. sequences or collections are expanded, operations are performed, fixed length non-negative integers are transformed, and code values are imported from the code tables.

This gives a list of simple features identifiers, and an equivalent list of values or code values to which they map. Coordinate groupings and collection groupings can give the tree hierarchy of features, and a further mapping can expand feature names, code value meanings, parameter units etc..

This is entirely equivalent to a full specification in a markup language, where the specific document schema travels with the document, like an internal, encapsulated DTD contrasted with an external, referenced DTD or an XSD schema.

3.4 BUFR UML

The WMO BUFR standard is described here with the aid of UML diagrams for the first time. These are only a high level description and do not show anything approaching the detail of the BUFR tables. There are around 450 BUFR tables, containing around 7000 rows of features, collection elements, codes and flags.

The first 7 diagrams show the BUFR model, while figures 8 to 12 illustrate BUFR coding mechanisms. In ISO terms, these should be separate, but the BUFR standard admixes them to provide what can be considered as “dynamic” models which can define new feature classifications in an instance (a BUFR message).

A simple description of UML connectors is to be found in Annex A and illustrated in Figure A.1. The full definition is maintained by the Object Management Group (OMG) (see reference 6.4).

3.4.1 BUFR model

Figure 1 shows the main package structure of the WMO BUFR Standard. This structure makes explicit the concepts which are often merged together in the BUFR documentation. It is an attempt to separate format and coding concepts from the data modelling concepts.

Figure 2 describes the structure of the Classification Tables which in ISO terms is the set of BUFR Feature Catalogues.

Figure 3 shows the full set of Table A sub-catalogues. This is the top-level classification of the BUFR catalogue.

Figure 4 lists the BUFR generalised coordinate system. In ISO terms this is a mixture of three functions: the first is a grouping mechanism by which features and coverages are linked, the second describes Coverages and coverage grid definitions and the third can be seen as a way to define feature attributes. Consequently BUFR coordinates are much more variable than ISO coverages. BUFR coordinates also allow an enormous flexibility in defining coverages of continuous or discrete variables.

Figure 5 explores the classification of simple BUFR features. BUFR has no explicit mechanism to define inheritance and association, but in practice the Table D defines sequences and groupings, which serves such functions in conjunction with the BUFR coordinate mechanisms. This is very flexible, but this flexibility also means that it is not possible to enumerate and define all potential BUFR Feature Catalogues.

Figure 6 expands the BUFR simple feature class for temperatureFeatures which are all continuous coverages.

Figure 7 picks on two simple features in the class of observedFeatures, “Type of precipitation” and “Ice development”. These are discrete coverages and show the detailed definition of the discrete values they can take in the BUFR code tables.

3.4.2 BUFR messages – coding of BUFR instances

Figure 8 illustrates the structure of a BUFR Message. This describes the logical structure at a level close to the physical structure, and is what most BUFR documentation starts with.

Figure 9 is a diagram of the four types elements of BUFR coding operators, one of which, the replication operator is described fully.

Figure 10 lists the BUFR coordinate operators, tightly allied to the coverage classes shown in Figure 3.

Figure 11 discusses the feature descriptor qualifiers. These are operators which perform some of the dynamic feature modification described in class 31 of Figure 4. As well as increased data compression, these operators allow the specification of dynamic coverage grids within a BUFR message or instance.

Figure 12 lists the coding operators of BUFR Table C which explicitly modify the coding “attributes” of any BUFR feature. These are operators which modify the coding, presentation or annotation of items or groups of items in a BUFR instance. They have no affect on the BUFR model directly, but they can add annotation or quality measures to a predefined feature.

3.4.3 BUFR Decoding

Figure 13 shows and Action Diagram describing the conversion of a BUFR message or instance to a basic form of XML, and indicates a route to populate a Feature Catalogue from enumeration of the feature types in a BUFR instance.

Figure 14 describes the process of decoding a BUFR message to an expanded form. This is little more than the current BUFR decoding process.

However the next steps, shown in Figure 15, give a route map to convert a BUFR message into an XML dialect; show that creation of a schema for this is straightforward, but redundant; and indicate how the feature types defined in the BUFR message might be captured and added to a Feature Catalogue.

4 BUFR UML description

4.1 Figure 1: the BUFR UML package overview.

In this view, some of the BUFR concepts are described in identifiable packages. The package structure is an attempt to separate out components from an ISO modelling viewpoint, but it is heavily influenced by actual BUFR structures.

4.1.1 BUFR top level packages

The UML should naturally split into data model (which mostly concerns feature catalogues) and the data instances. However BUFR is a data-driven design, where the data instance (**BUFR_Message,** figure 8) contains the data and the schema (**BUFR_Model**) defining the data, but also operations which can define (**BUFR_Operations,** figure 9) new and previously unspecified feature types with new feature attributes.

4.1.2 BUFR_Model::Classification Tables

This package is expanded in Figure 2, and explains the linkages between Tables A (BUFR_featureCatalogue, shown in Figure 3), Table D, elements of Table B, code tables and some of the Common Code Tables. Table D lists collections of features (and collections of collections). There are 300 such collections with 4000 elements in them, and Table B contains 1200 simple features in 30 classes.

4.1.3 BUFR_TableElements

The classes for this package include all the Table B classes of simple features and are described in Figure 5. We separate out the **BUFR_coordinateOperators** (figure 4) which are also included in Table B. Figure 6 describes one element from one class table for temperatureFeatures.

4.1.4 Code Tables

Figure 6 describes one element of a coverage class of Table B of a continuous variable, and Figure 7 shows two elements of a coverage class of discrete Observed Phenomena. There is no practical way to show all 1200 Table B features, 370 of which have enumerations - code or flag tables, containing around 3000 codes and flag values

4.1.5 BUFR_Operations

BUFR_Operators are described in figure 9. Operators split into two types, coding operators (**CodingMethods**) and model modifiers (**BUFR_coordinateOperators**). **PackingMethods** are also CodingMethods, but the operator is handled differently in BUFR and is part of the **BUFR_Message** structure, so it is packaged separately.

4.1.6 BUFR_coordinateOperators

Table B Coordinate descriptors are BUFR mechanisms to declare coverage grids, and more generally they are also BUFR's mechanism to define a limited tree structure, and contain grouping elements and feature attribute definitions. The linkages in this

package are described in Figure 4, and some more detail on the operators is shown in Figure 10.

4.1.7 DescriptorQualifiers

DescriptorQualifiers can also be seen as operators, but the are both coding and model operators, and they are also a Table B sub-class. With 3 potential associations in Figure 1, it is necessary to treat the **DescriptorQualifier** class as a special case. They are described in figure 11.

4.1.8 CodingMethods

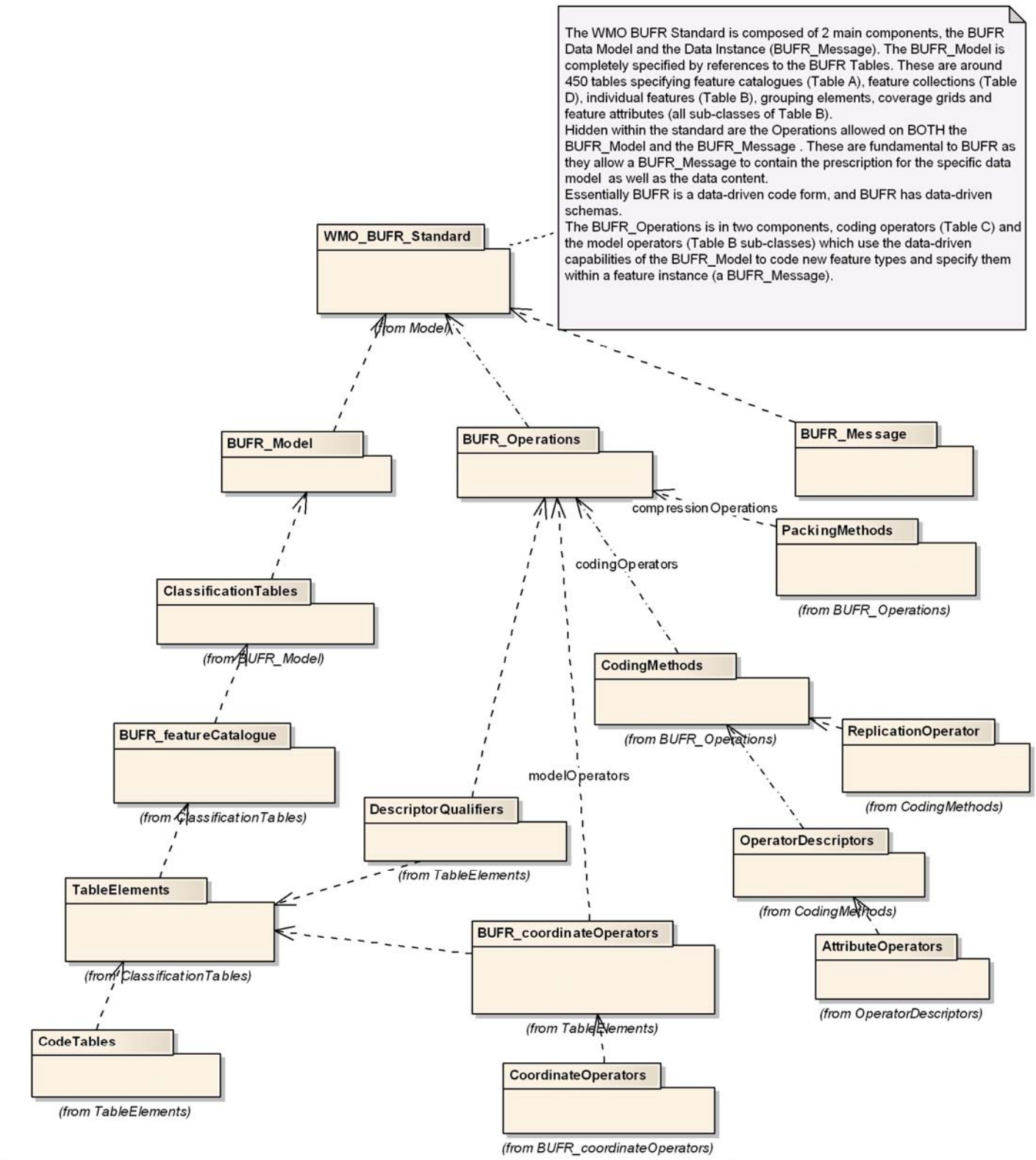
The coding operators are described in Figure 12. These are the Table C operations (without the Quality Assessment function, see 2.3).

4.1.9 Packing Methods

This refers to the extra packing processing which can be used. It is shown in Figure 12, bit it is not explained in this paper, as this is a coding issue.

4.1.10 BUFR Message

Figure 8 shows the components of a BUFR message – or feature instance in ISO parlance. The diagrams of section 5 (figures 13, 14 and 15) show the outline activities needed to expand an individual message into XML and how the embedded data model can be created to contribute to a static feature catalogue



Within the BUFR_Model, the ClassificationTables contain all the BUFR information. (Here the ClassificationTables are being taken to exclude the Table C codingOperators).
The BUFR_featureCatalogue is described in 20 sub-catalogues defined in Table A. TableElements include Table D sequences (300 sequences with 4000 elements), which are pre-defined collections of other Table D collections and simple BUFR features of Table B (30 classes with 1200 elements). One sub-class of Table B are the replication operators (DescriptorQualifiers) and simple features have either fixed point numbers or code table values as content. There are 370 code tables, flag tables and enumerations.
However 10 of the Table B sub-classes define featureModifiers which BUFR describes as generalised "co-ordinates" which are actually operators which give BUFR its data-driven nature. The CoordinateOperators deliver the ISO equivalent of grouping elements in a hierarchy, elements specifying coverage grids, and the expression of feature attributes.

Figure 1: WMO BUFR Standard UML Package

4.2 Figure 2: BUFR feature catalogue

This is perhaps the most important diagram of this paper. It summarises the BUFR model and the relationships behind the BUFR tables.

4.2.1 Feature

The ISO 19100 series standards define a "feature" as an "abstraction of a real world phenomenon". BUFR tables are lists of features, and ways in which these features can be further defined. In ISO terms these further definitions can be child features (secondary phenomena), feature attributes (properties of features) and feature associations (relationships between features).

A feature catalogue is a full description of feature types, and a feature instance is an example of a particular feature or feature set. For BUFR, the feature catalogue is equivalent to the BUFR tables and the relationships between the tables, and the feature instances are BUFR messages.

However, the BUFR model allows for features declared in the BUFR tables to be redefined dynamically. This happens in at least 4 distinct ways. The redefinition mechanism is so flexible that it cannot easily be enumerated, and BUFR leaves the definition open to the data producer.

4.2.2 Coverage

In ISO terms a "coverage" is a type of real world phenomena that varies over position. It is expressed as a set of coverage pairs, a position in the spatio-temporal domain, and a record of values, each value in the record associated with an elements measured at the position in an array of points or cells. The geographic paradigm usually associates coverages with regular grids, and features with vector data, but this is too restrictive for the BUFR concept of "coordinates".

ISO adopted the term "coverage" from the Abstract Specification of the Open GIS Consortium, to refer to any data representation that assigns values directly to spatial position.

In meteorology, since nearly everything we measure is a coverage, we need, at the minimum, to extend the domain of the coverage grid from the strictly spatial, to the spatio-temporal and then to the spatio-temporal-parametric domains[11]. A useful distinction is that a feature may have an identifier or name (such as a SYNOP, or a tropical cyclone) while the elements in the SYNOP are all coverages such as temperature at the location.

To illustrate:
It is common to visualise the cloud amount at a specific height over a regular grid in a region. Here the coverage grid is the horizontal grid, the height value and the time at which it is

[11] This is covered by 19111-2, Coordinate Reference Systems using a parametric coordinate axis.

measured. The coverage is the cloud amount measured at all points in the coverage grid. The fixed height values are the independent parameter and the variable cloud amount is the dependent parameter
But equally, to show a pilot the limits of Visual Flying Rules, we might visualise the height at which 3/8^{ths} or 5/8^{ths} of cloud occurs at all points in the horizontal grid at that time. In this case the coverage grid is the horizontal grid, the time it was measured, and the point in the parametric coordinate – cloud amount. The cloud amounts represent the independent parameter and the height, the coverage variable, is the dependent parameter.

4.2.3 Limitations of the concept of Coverage for Meteorology.

The question arises – is the extension of Coverage to the spatio-temporal-parametric domain enough for the multi-dimensional atmospheric or oceanographic regime?

By multi-dimensional, we mean more than 3D spatial and time, we intend that the parametric dimensions be included – dimensions in the mathematical sense. In an atmospheric model, at each 4D spatio-temporal location we define many parameters: e.g. temperature, pressure, humidity, 3 winds dimensions, physical-chemical constituents (liquid and solid water, aerosols, perhaps many chemical abundances and rates), fluxes (radiation, heat, fluid) etc..

Many visualisations and data extractions can be derived by projecting this high dimensional model into lower dimensions, using coordinate axes and CRSs which need not be spatial but parametric (e.g. cloud amount as an axis with 5/8^{ths} as a coordinate).

ISO 19123 “Geographic information – Schema for coverage geometry and functions” is the ISO/TC211 standard which applies to coverages.

In this coverages are defined as mappings of values to positions, with a defined set of simple operations such as listing and selecting, but also interpolations and evaluations and inversions (mapping from point to value and vice versa - contour production from the grid being offered as an example of an inverse evaluation). The creation of a coverage grid is a distinct task in addition to coverage mappings.

Then much of the definition of the standard is taken up with grid definitions which are relevant to 2 dimensional GIS techniques, where direct interpolation from the raw values is the norm. Hierarchies of discrete and continuous coverages are described which, though it mentions solids and surfaces, is essentially 2D in nature. It has little concept of projections from a multidimensional tensor space.

Where it discusses the distinction between discrete and continuous coverages, it distinguishes between a mathematical mapping between position and value as “analytic” continuous coverages, but recognises that continuous coverages are often represented as a discrete “control” coverage on a grid of points. Continuous coverages are those which can be interpolated. Here is the only recognition of a possible multidimensional nature, as the standard recognises that the geometry in which the interpolation would be performed may be of a “higher topological dimension”

This seems a very minimal recognition of projections from multidimensional models, discontinuities such as fronts, troughs, confluences and diffluences or mathematical poles such as cyclone centres. It does not consider piecewise continuous coverages such as rainfall amount in a period.

4.2.4 BUFR treatment of coverages.

The BUFR coordinate specification shows that collections, coverages and attribute assignment are all similar functions, which BUFR treats identically as a grouping function.

While ISO coverages imply that operations on coverages should be declared (list, select, evaluate inverse etc.) BUFR only has mechanisms to define coding operations, and to declare operations performed in the creation of the data. It does not define operations which can be performed on the data.

However, rather against the intention described in 2.1, to make the obvious explicit, the whole status of atmospheric modelling is implied. Many of the BUFR “coordinates” have operations dependent on sophisticated atmospheric modelling, rather than only the simple operations defined for ISO coverages.

This is a spacing page.

4.2.5 BUFR feature Catalogue

The top superclass in figure 2 is the BUFR_featureCatalogue. This is the generalisation of the BUFR_specialisedFeatureCatalogues which are realised by the list in table A, expanded with Common_codeTable_C-13. This list is described in Figure 3.

A BUFR_specialisedFeatureCatalogue is composed of BUFR_table_D_sequenceClasses although there is no requirement to use Table D sequences in a BUFR message or instance and data creators can express the feature set in Table B descriptors only.

A BUFR_table_D_sequenceClass is realised by the BUFR Table D. In ISO terms this is a feature collection. A feature collection is also a feature. This recursive property is modelled by composition connecting BUFR_table_D_sequenceClass back to itself. A Table D sequence class can contain other sequence classes. The only rule is that there can be no self-recursion – a BUFR_table_D_sequenceClass cannot include itself.

Ultimately, all BUFR_table_D_sequenceClass-es (Table D collections) are composed of Table B elements, BUFR_table_B_simpleFeatures, realised obviously by Table B references. This is modelled in Figure 5 and an example simple feature is shown in Figure 6

However many Table B references are made up of special code tables, if they are not represented by measures of the parameter. Here these are described as Table_B enumerations, discreteCoverageTables and flagTables. BUFR just calls them code or flag tables, and 370 of the 1200 BUFR_table_B_simpleFeatures have code/flag tables. Two examples are shown in Figure 7.

On the BUFR_table_B_simpleFeature class there is a recursive dependency. This shows that some Table B features (more than the coordinate classes) modify other Table B features, and one class of Table B features which does this is class 31 – “Data descriptor qualifier operators”.

The other operator functions of Table B are the coordinateOperators classes. These are subclasses of BUFR_table_B_simpleFeature but are modelled as components of the BUFR_modelOperator class. The coordinateOperators (which BUFR calls generalised “coordinates”) are the BUFR classes from 0 to 9 (although 3 and 9 are reserved classes with no members). This is described in more detail in Figure 4.

The BUFR generalised coordinates should really be considered as operators rather than features. As described in 4.2.4 and later in 4.4 and 4.9, these operators perform multiple processes in ISO terms. The operators as depicted in the UML diagram of Figure 2 as information “flows” which modify the BUFR_table_D_sequenceClass and BUFR_table_B_simpleFeature classes.

These featureModifiers are the essence of the data-driven character of BUFR. There are at least 1500 featureModifiers through the classes and coded values of the coordinateOperators which can operate on the other BUFR_table_B_simpleFeatures in single mode. Many can behave as multiple operations, giving many million potential combinations.

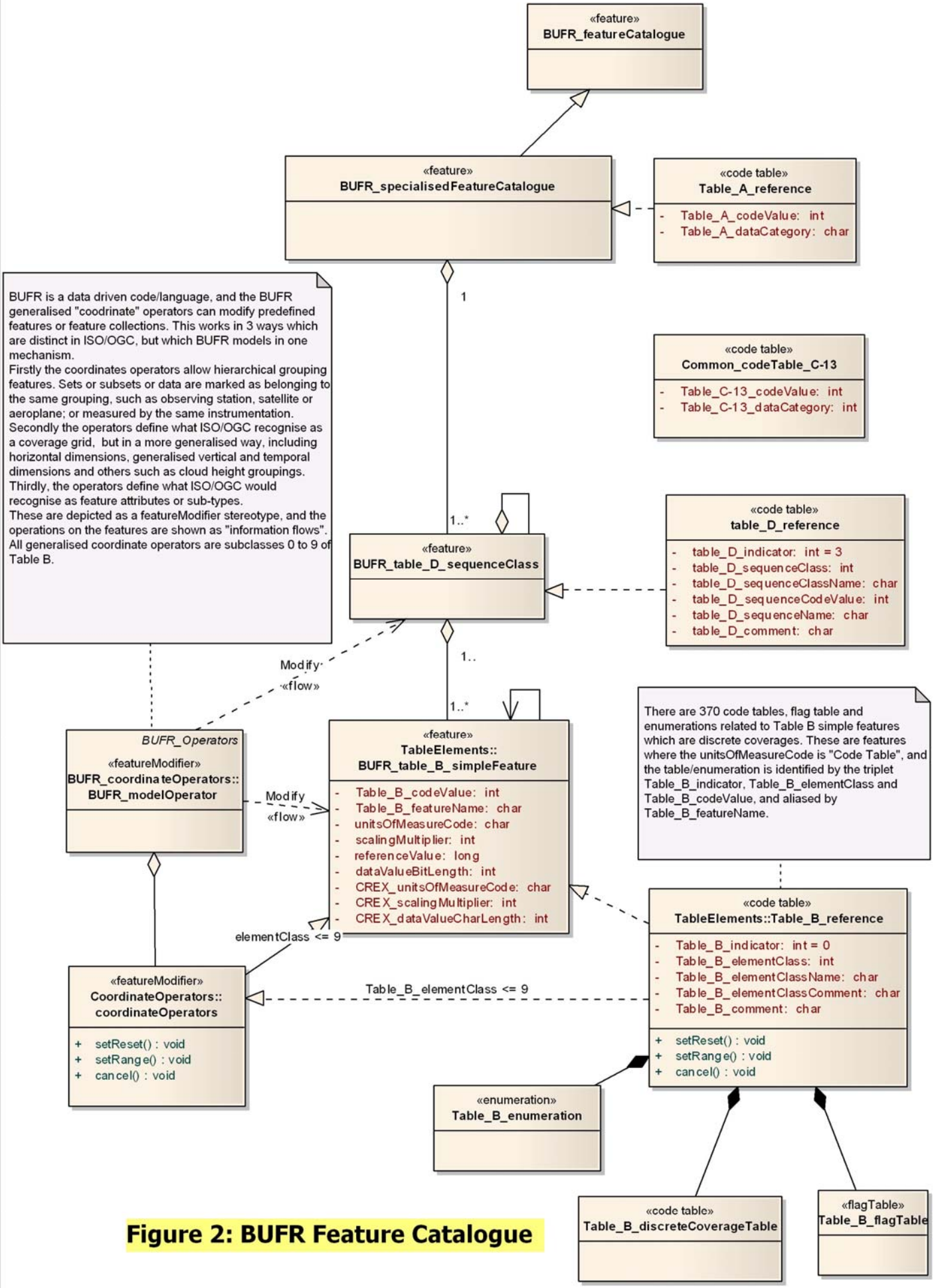


Figure 2: BUFR Feature Catalogue

4.3 Figure 3: Table A Data Categories

There are 20 sub-classes of Table A which represent sub-catalogues of the BUFR model.

The specialisation of the BUFR_featureCatalogue into BUFR_specialisedFeatureCatalogues is not really a fundamental part of the BUFR model. Many of the component classes of BUFR_table_D_sequenceClasses (Table D collections) and the classes of Table B elements of BUFR_table_B_simpleFeature are used and reused as modules of other feature collections and catalogues. So there are no one-to-one relationships with Table D and B classes and Table B sub-catalogues. There is some attempt to use the classification of Table A catalogues in the classification of Table D, but this works best for some of the more esoteric classes, which do not see much re-use of the component features.

Some of the classifications reflect the way in which the data is gathered, rather than any structural feature. So surface data is split between land and sea (and satellite observations assigned to the surface). Vertical data distinguishes between satellite and standard (non-satellite) soundings, and is further distinguished between single layer data and soundings through the depth of the atmosphere. At least part of this is to recognise that aircraft measurements keep mainly to single layers except at take-off and landing. Part, though is that some satellite products measure parameters at a nominal layer or band, rather than a sounding.

Others are product data: warnings, forecasts, status information, and synoptic features.

Although this set of sub-catalogues are part of the high-level description in BUFR, they are not particularly well utilised in the BUFR model. However it is indicative that the BUFR model encompasses quite a range of specialised user requirements.

In section 5 a route-map to convert BUFR messages into a GML application language. The section explains that expressing the whole BUFR model in a feature catalogue is not possible because it is too big. However the existence of the BUFR_specialisedFeatureCatalogues is a strong indication that there are specialist and non-specialist communities of users who could benefit from explicit sub-catalogues recast under an ISO model. Section 5 proposes a process to develop ISO feature catalogues as sub-catalogues of BUFR.

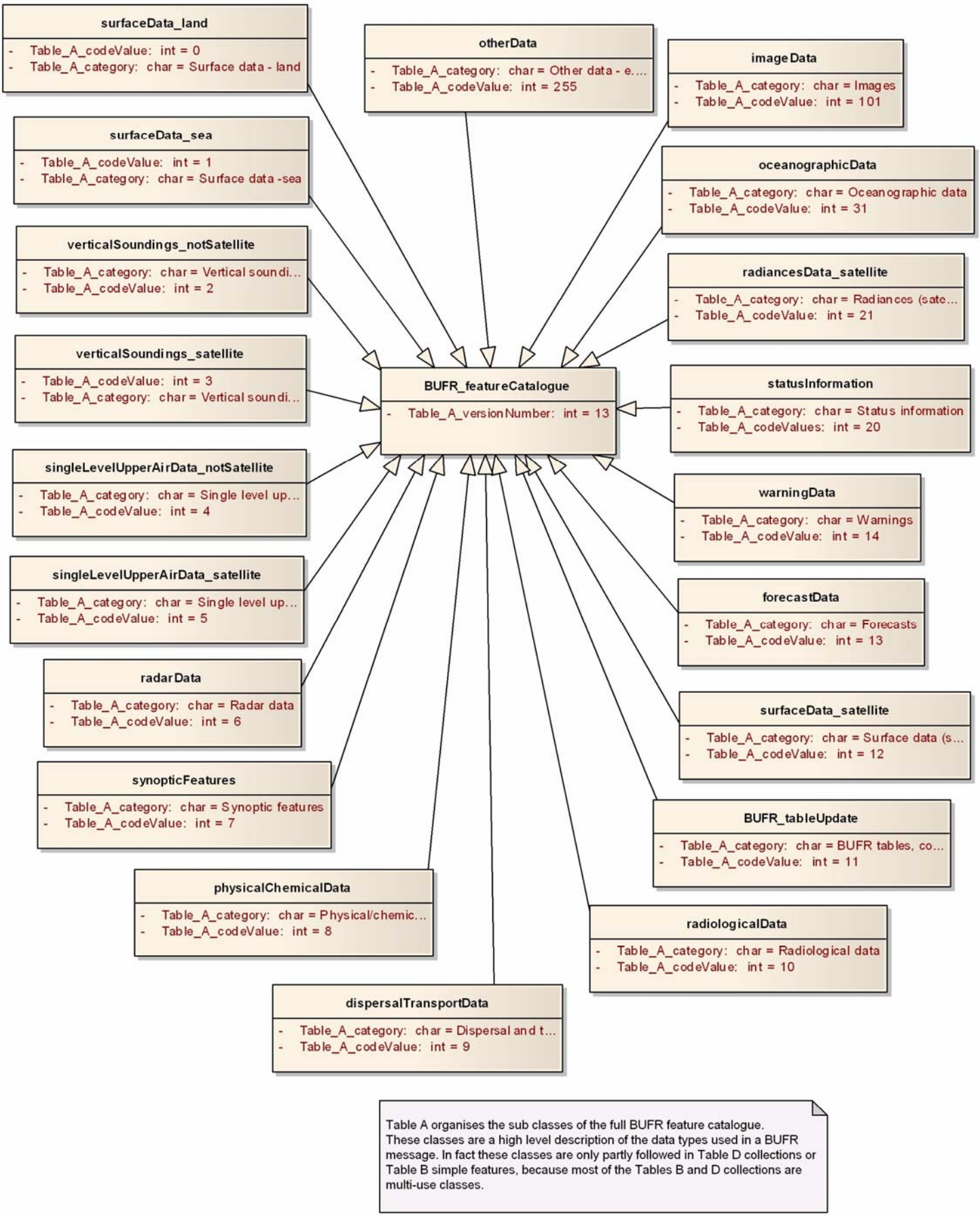


Figure 3: Table A Data Categories Feature sub-catalogues

This is a spacing page.

4.4 Figure 4: BUFR coordinates – the BUFR coverage specification

The first three classes of Figure 4 replicate part of the BUFR feature catalogue in Figure 2.

BUFR coordinates aren't restricted to spatial grids. In fact they aren't particularly good for large multidimensional regular grids – GRIB is designed for that. BUFR coordinates are primarily a mechanism both to group sub-features and to define coverage grids, in either case to contain features or feature collections.

The Table_B_reference is the realisation of the BUFR_table_B_simpleFeature class. The special code tables (enumerations, discreteCoverageTables and flagTables), although not shown in the diagram, also have their use in the BUFR definition of coverages. The range of the coverage grids, particularly for significanceQualifiers, is defined there.

The recursive dependency on the BUFR_table_B_simpleFeature class probably does not have a use in the coverage grid, but the functionality (some Table B features modifying other Table B features) is available in the BUFR model – should anyone come up with a use for it.

The coordinateOperators classes are a subclass of BUFR_table_B_simpleFeature. While the coordinate descriptors are the BUFR classes from 0 to 9, in practice classes 3 and 9 are reserved and empty classes with no members.

4.4.1 Grouping mechanism to specify BUFR Tables

Since the BUFR tables can be exchanged within a BUFR message, either as complete tables or only updates, it isn't surprising that there is a grouping mechanism (Class 0) for specifying BUFR tables.

4.4.2 Spatio-temporal and "vertical" coordinates

Spatio-temporal coverage grids can be defined with classes 4 to 7.
Class 4 defines temporal location, and it can define absolute values, increments, periods or time displacements, durations or local time displacements.
Classes 5 and 6 are called 1st and 2nd horizontal location. These aren't necessarily distance measures either, but can be angles of elevation and azimuth; rotation and radius; or along and cross track positions for a ship, aeroplane or satellite.

Class 7 is the "vertical" parameterisation, which is not often height, but geopotential, pressure, depth, water pressure or density.

4.4.3 Identifiers and Instrumentation

Class 1 is an "identifier" coordinate. While this is often related to the position of an observing station, it is also the identifier for a moving station, such as an aircraft, or a moving feature, such as a tropical cyclone. Here positions are attributes of the observation, not a BUFR "coordinate", coverage or otherwise. In ISO terms, this would be inverted, and the position-record value pairings would be grouped in a "CV_DiscreteCurveCoverage".

In practical terms, both coding and modelling, BUFR make no such distinction.

Class 2 is a “coordinate” describing the instrumentation used to measure the feature values. This is used by the instruments and observing programme of WWW, but is also used in the monitoring and verification programmes. This illustrates another difference in viewpoint from the ISO/TC211 which emphasises position above other attributes of the data, but it also represents a different outlook from other parts of WWW, such as the forecast production programme which emphasises the information in the observation values.

4.4.4 BUFR significance qualifiers

Class 8 illustrates, in the same class, both major similarities and major distinctions between BUFR coordinates and ISO coverages.

Class 8 groups the following data by significance qualifiers, along with Table B class 31 (Figure 11) and other appropriate BUFR coordinates, such as vertical pressure measures, these assign local coverage grids for special observations like a vertical sounding with a variable number of levels. The Table B delayed replications allows the number of features and replications to be specified in an instance, the significance qualifier will define what sort of level it is, the second coordinate type will assign a value to the level, and the remaining features will specify values of the features at those levels.

This type of qualifier can be regarded, in ISO terms, as a feature attribute mechanism, but this is a dynamic mechanism, and may apply to many feature sets.

However other class 8 significance qualifiers declare the following features to be summary statistics of a data set, in time or space, which is declared in the BUFR instance. Declaring the following features to be first order statistics, difference statistics, or to qualify the values for missing data etc. is effectively a major re-definition of the simple features in Table B.

All but a few of 50 or so sets of class 8 features are realised by code tables specifying attributes of the following set of features. Expanded up to count code table values there are nearly 1000 individual significance qualifiers.

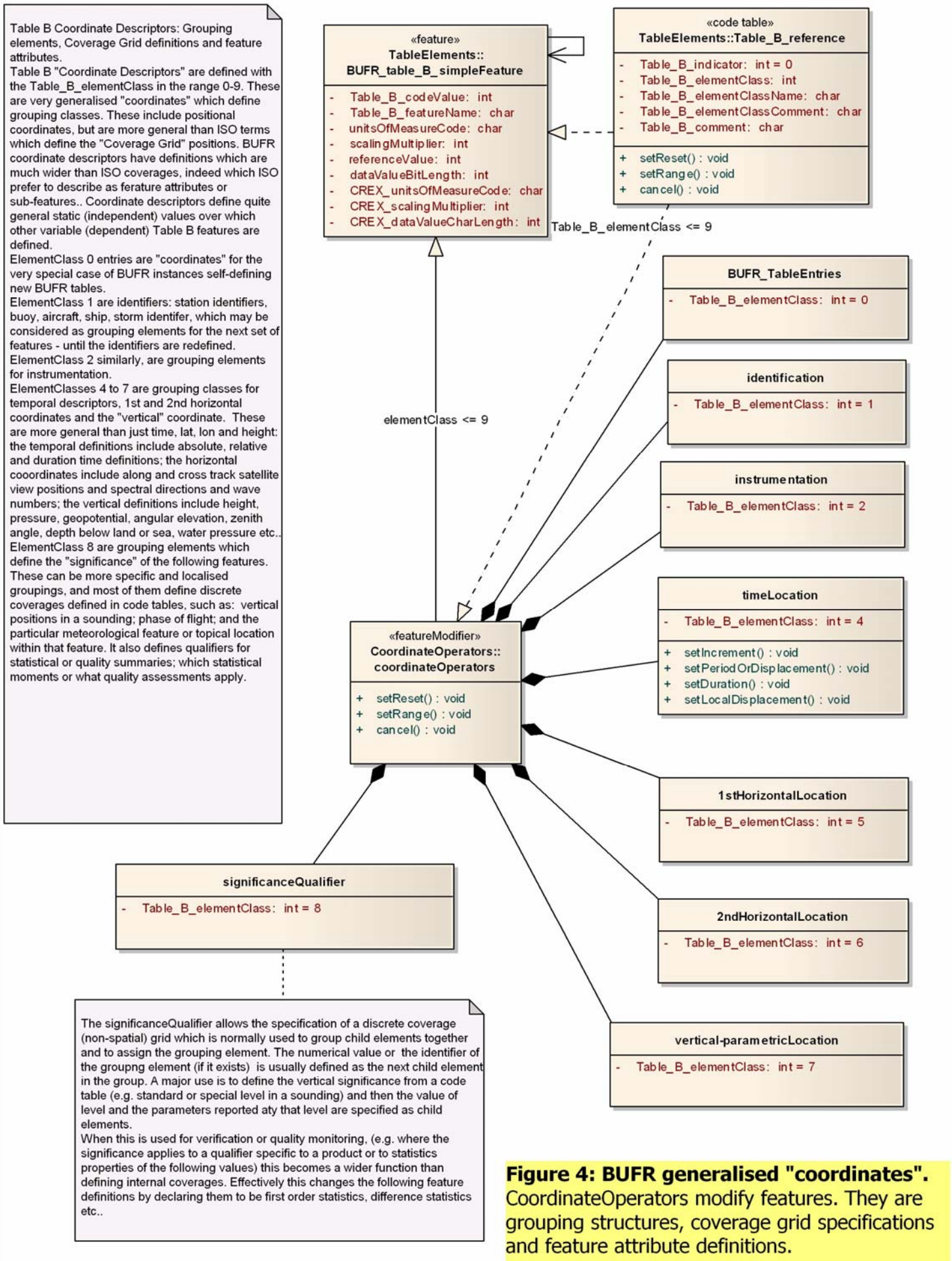


Figure 4: BUFR generalised "coordinates". CoordinateOperators modify features. They are grouping structures, coverage grid specifications and feature attribute definitions.

4.5 Figure 5: Table B simple Features

BUFR_table_B_simpleFeatureTypes are features which are realised by the Table B. Those classes of Table B which are BUFR coordinates are not listed here, as they are modelled in Figure 4.

The feature class is composed of 20 sub-classes of features, some of which reflect the subject classes of Table A and D, others not. The classification of individual features is quite tightly bound to the structure of Table B classes.

The recursive association in BUFR_table_B_simpleFeatureTypes represents the class of Table B where features in Table B class 31 operate on other table B features or sets of features. This is shown in more detail in Figure 11. However it also refers to functions of some other tables (such as Table_B_elementClass 25, describing processingFeatures) which allow a table reference to modify the context of the succeeding table reference. While these can be considered a restricted operator they are not considered further.

While the coordinateOperators of Figure 4 are specified as separate classes to the simple feature types, when these feature types are not used as coordinates, they are replicated as feature types (see footnote to 3.3.2). The set of classes are described as nonCoordinateLocations, and include vertical features, temporal features and 1st and 2nd non-coordinate horizontal locations.

As an example, the temperatureFeatures (Table_B_elementClass = 12) is described in more detail in Figure 6 as an example of a continuous coverage. Two examples of discrete coverages are described in Figure 7 for observedPhenomena (Table_B_elementClass = 20).

There is no capability for multiple inheritance in BUFR, so it is obvious that features which exhibit properties of more than one class have usually been assigned to a single class. While the maintainers of BUFR have tried to limit multiple instances of what is effectively the same feature, there are examples where limiting BUFR features to single inheritance seems to be constraining. The maintainers are quite happy, on the other hand, to permit denormalisation for presentational purposes, for example, there are two sets of identical features in the temperature class, where the difference is in the precision of the feature. This sort of denormalisation is accepted for the purpose of space efficiency.

The comments in 4.6.2 on implicit structures within classes indicates where the practical limitation of allowing BUFR only one level of classification of tables, is a constraint on hierarchical modelling.

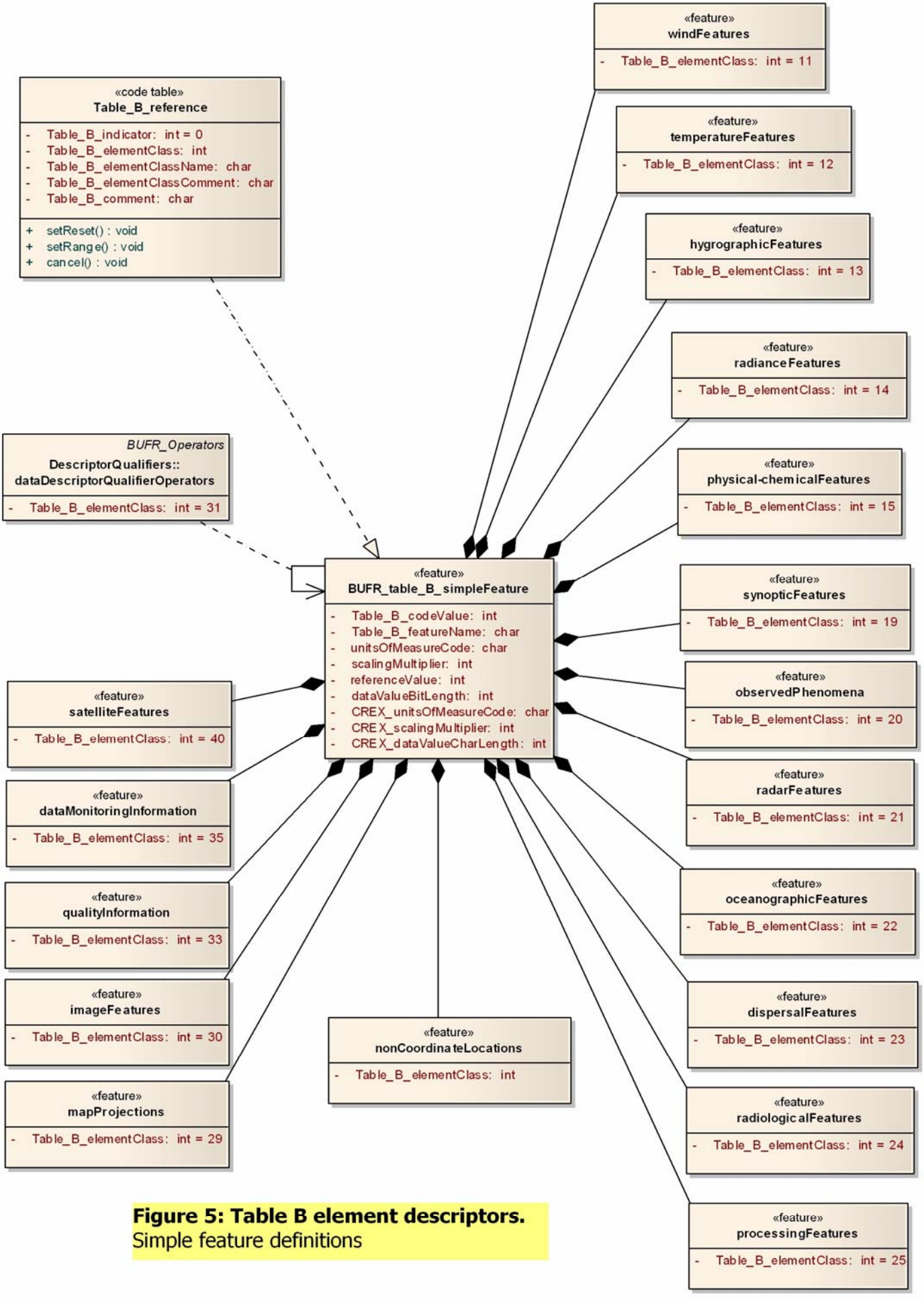


Figure 5: Table B element descriptors.
Simple feature definitions

4.6 Figure 6: Temperature feature code table example of a continuous coverage

The feature class temperatureFeatures is a component of the BUFR_table_B_simpleFeature, and the 2mDewPointTemperature (Table_B_codeValue = 6) is a specialisation of that general class. The temperature example illustrates that BUFR is detailed rather than complex.

Part of the class of temperatureFeatures is listed in Figure 6 as an annotation to the UML. A small subset of the elements (rows) are shown.

4.6.1 Number representations in BUFR

This annotation can be used to explain the specification of the number precision mechanism.

The first 3 columns are the expression of the index. The element name makes up the element identifier, and then the BUFR and CREX attributes are listed.

For the Dew-point temperature at 2 m (Table_B_codeValue = 6), the units in a BUFR message are expressed as Kelvin. The scale is one, so the value is multiplied by 10^1 before rounding to integer (and so has a precision of 0.1 K). There is no reference offset, and the unsigned integer is stored in 12 bits. This means that the lowest value possible Dew-point temperature to be stored in a BUFR message is 0.0 K and the maximum is 409.6 K.
These extremes which can never occur also show how further data compression can work. If the lowest Dew-point temperature in an array of temperatures was 253.0 and the maximum was 303.0, a further reference value of 2330 could be removed from the array, and the range 0 to 500 could be stored in 9 bits per array entry.

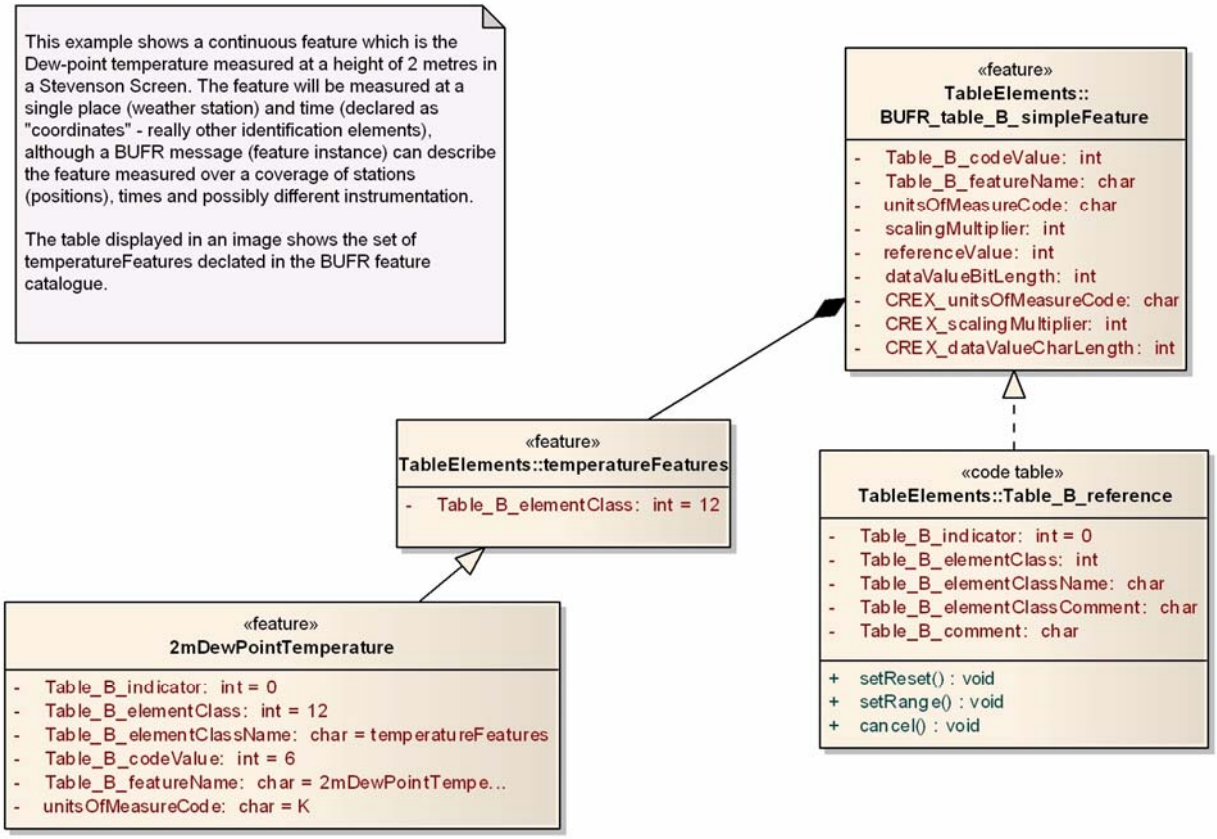
4.6.2 Implicit structures in the temperatureFeature table.

While the data precision for 2mDewPointTemperature (Table_B_codeValue = 6) has a precision of 0.1 K, a higher precision version exists in element Table_B_codeValue = 106. This is stored in 16 bits and has a numeric precision of 0.01 K.

Storing different precisions of the same feature as different elements of the class is a device used by BUFR which ISO would assign as a presentation property, only marginally connected with data modelling. This denormalisation device is one of several which are used to gain format efficiencies.

An ISO feature catalogue might also treat dewPointTemperature (element 3) and 2mDewPointTemperature (element 6) as the same feature with different attributes (element 3 would have a level specification in a coverage grid in the vertical, while the level attribute for element 6 would be 2metres).
Similarly the minimum and maximum temperatures might be temperatures with specific attributes of maximum or minimum, also at a specific height and over a specific period.

In sub-classing temperature, it might also be thought sensible to model radiation temperatures, skin temperatures and brightness temperatures as specialisations of temperatureFeature which are quite distinct from thermometric temperatures.



		Table_B_elementClass = 12 Table_B_elementClassName = temperatureFeatures	
F=Table_B_indicator	Table_B_featureName	UNIT = unitsOfMeasureCode	UNIT = CREX_unitsOfMeasureCode
X=Table_B_elementClass		SCALE = scalingMultiplier	SCALE = CREX_scalingMultiplier
Y=Table_B_codeValue		REFERENCE VALUE = referenceValue	REFERENCE VALUE = CREX_referenceValue
		DATA WIDTH (Bits) = dataValueBitLength	DATA WIDTH (Characters) = CREX_dataValueCharLength

TABLE REFERENCE			TABLE ELEMENT NAME	BUFR				CREX		
F	X	Y		UNIT	SCALE	REFERENCE VALUE	DATA WIDTH (Bits)	UNIT	SCALE	DATA WIDTH (Characters)
0	12	001	Temperature/dry-bulb temperature	K	1	0	12	°C	1	3
0	12	002	Wet-bulb temperature	K	1	0	12	°C	1	3
0	12	003	Dew-point temperature	K	1	0	12	°C	1	3
0	12	004	Dry-bulb temperature at 2 m	K	1	0	12	°C	1	3
0	12	005	Wet-bulb temperature at 2 m	K	1	0	12	°C	1	3
0	12	006	Dew-point temperature at 2 m	K	1	0	12	°C	1	3
0	12	007	Virtual temperature	K	1	0	12	°C	1	3
0	12	011	Maximum temperature, at height and over period specified	K	1	0	12	°C	1	3
0	12	012	Minimum temperature, at height and over period specified	K	1	0	12	°C	1	3
0	12	013	Ground minimum temperature, past 12 hours	K	1	0	12	°C	1	3
0	12	014	Maximum temperature at 2 m, past 12 hours	K	1	0	12	°C	1	3
0	12	015	Minimum temperature at 2 m, past 12 hours	K	1	0	12	°C	1	3
0	12	016	Maximum temperature at 2 m, past 24 hours	K	1	0	12	°C	1	3
0	12	017	Minimum temperature at 2 m, past 24 hours	K	1	0	12	°C	1	3
0	12	021	Maximum temperature at 2m	K	2	0	16	°C	2	4
0	12	022	Minimum temperature at 2m	K	2	0	16	°C	2	4
0	12	030	Soil temperature	K	1	0	12	°C	1	3
0	12	049	Temperature change over specified period	K	0	-30	6	°C	0	2
0	12	051	Standard deviation temperature	K	1	0	10	°C	1	3
0	12	052	Highest daily mean temperature	K	1	0	12	°C	1	3
0	12	053	Lowest daily mean temperature	K	1	0	12	°C	1	3
0	12	061	Skin temperature	K	1	0	12	°C	1	3
0	12	062	Equivalent black body temperature	K	1	0	12	°C	1	3
0	12	063	Brightness temperature	K	1	0	12	°C	1	3
0	12	064	Instrument temperature	K	1	0	12	K	1	4
0	12	065	Standard deviation brightness temperature	K	1	0	12	K	1	4

Figure 6: Temperature feature class.
An example of a continuous coverage.

4.7 Figure 7: Observed Phenomenon code table example of a discrete coverage

The feature class observedPhenomena is a component of the BUFR_table_B_simpleFeature with a Table_B_elementClass value of 20. The two features typeOfPrecipitation (which has a Table_B_codeValue = 21) and iceDevelopment (Table_B_codeValue = 37) are both specialisations of the observedPhenomena general class.

The unitsOfMeasure attribute for both features have the value “code table”, so the features are realised by code tables which are referenced by the index of their table B element.

The typeOfPrecipitation element has an index 0-20-21 and the flag table “codeTable_0-20-21” realises the feature. This “codeTable_0-20-21” has a length of 30 bits. The annotation attached to the flag table shows all the flags. Since at least some of the types of precipitation, whether drizzle, rain, snow grains, diamond dust etc. are considered possible to occur at the same time, the observer can record as many as necessary within the same coded value.

The iceDevelopment element has an index of 0-20-37 and code table “codeTable_0-20-37” realises the feature. The annotation attached to the code table shows the possible values of the code, and the meaning. This feature represents the state of sea ice reported within a ship observation near or within an ice sheet, and the codes reflect all the potential classes that the ship observer may see.

4.7.1 Flag tables and Code tables

The distinction between flag tables and code tables is that flag tables allow multiple values to be valid at the same time (and coded as a bit in a bitSequence) while code tables record a single occurrence in an event. Code values are usually recorded only once within a single weather event (an “observation”) whereas flag values can include multiple descriptions valid for one observation.

Usually a code table implies a hierarchy within the values. Either there is only one possible value that can be observed, or the observer is instructed how to choose a single value. Either the most appropriate value will be chosen, or more likely the highest code value (indicating an inherent or perceived ranking in the varieties of the feature) will be recorded.

The BUFR distinction between flag tables and code tables is represented in XML with a cardinality value. When code/flag tables are represented in an XML dialect, the loading within a message will be an important implementation issue. Keeping the single flag value of a flag table will simplify the coding, but at some point the XML will have to be expanded to include all the conditions that the flag indicates for presentation to a user.

There are 1200 Table B elements and 370 of these have code of flag tables. All the tables are available in a form similar to the annotations at the WMO web site which is available in the references of section 7.

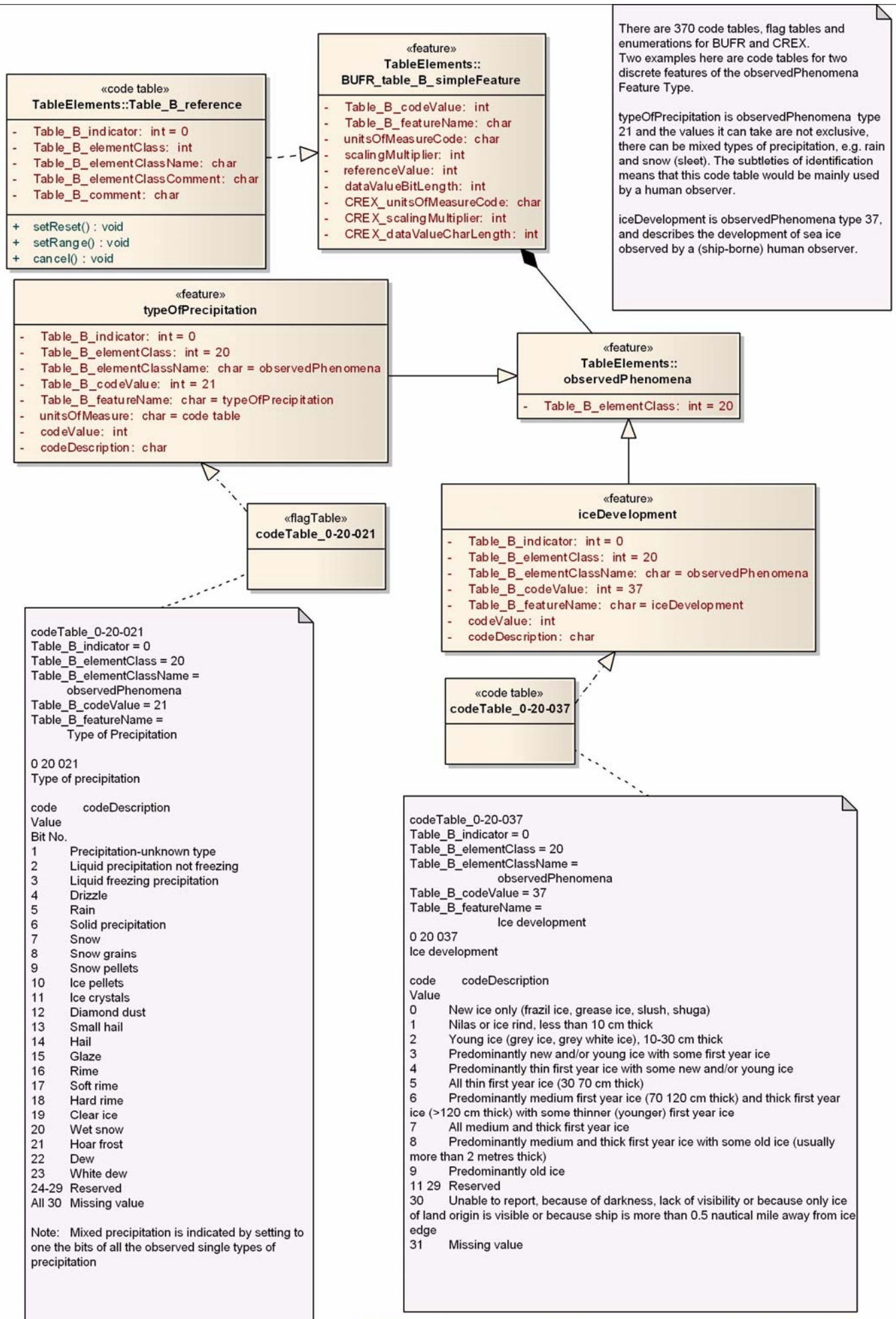


Figure 7: Observed Phenomenon code tables. Two examples of a discrete coverage.

4.8 Figure 8: The BUFR message – the feature instance

Figure 8 represents the structure of a BUFR instance – a BUFR message or bulletin sent between WMO data centres.

The structure of a BUFR message is normally what is discussed first in BUFR documentation. This reflects the BUFR designers' view, 20 years ago, that the coding was what was important and what the users wanted to know. The model and the BUFR tables were quite secondary. This paper takes a different view, and considers the BUFR tables, the whole modelling process, and the active maintenance of the standard to be a huge achievement.

This approach is complicated by the fact that a BUFR_Message contains its own schema (after referring to the tables) but also there are ways to use the BUFR_modelOperators to define new features (or at least feature modifications which haven't been specified before).

Figure 8 is organised slightly differently from other BUFR descriptions of the BUFR code form. These split the code form into 6 sections. Here the code form is considered to have 3 top level components, the BUFR_Metadata, the BUFR_dataDescriptionSection3 and the BUFR_dataSection4.

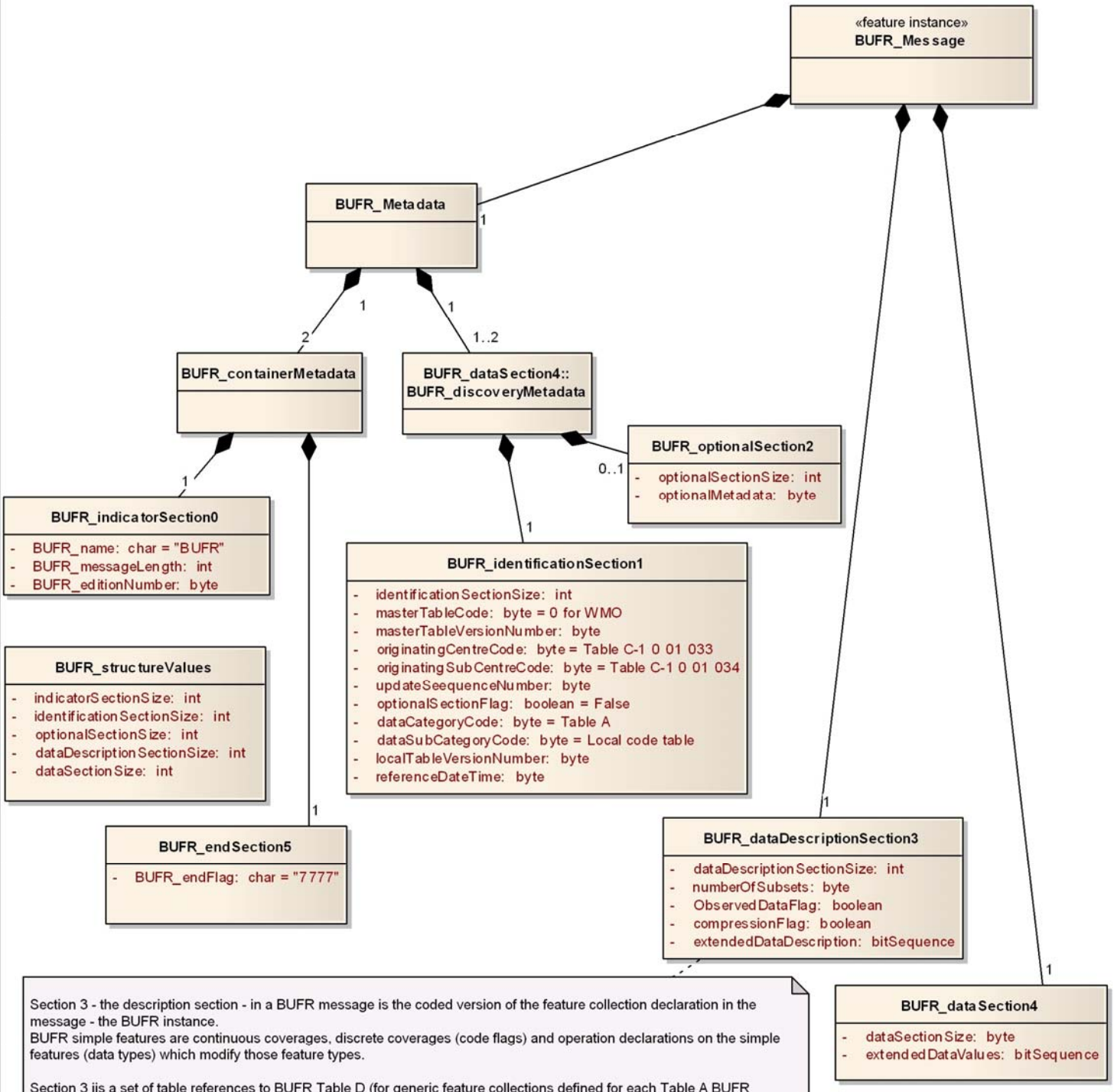
The BUFR_Metadata section contains two sections, BUFR_containerMetadata and BUFR_disoveryMetadata.

The BUFR_containerMetadata is the box structure within which the information is stored. The BUFR_indicatorSection0 has the BUFR identifier, the total length in bytes, and a version number. The BUFR_endSection5 has the end marker "7777". The three values (the identifier, the length and the end marker) together are a check on the integrity of the BUFR box. Each of the BUFR sections has a SectionSize contained in each section. These are all container information, not data, and are listed in the not-connected class BUFR_structureValues.

The BUFR_discoveryMetadata has two component classes, the BUFR_identificationSection1 and the BUFR_optionalSection2.
The BUFR_identificationSection1 has discovery metadata for the data contained within the message. Befitting the BUFR philosophy, these are often references to WMO code tables (Common Code Tables). However, some of what ISO might call metadata is contained within as data.

The BUFR_optionalSection2 is a very interesting addition. Normally it is unused. It is also only ever locally defined, and some repositories which use BUFR as the file form store database indices and foreign key information here as part of the repository structure.

The BUFR_dataDescriptionSection3 and BUFR_dataSection4 store the compressed, packed element identifiers and numeric values and codes respectively.



Section 3 - the description section - in a BUFR message is the coded version of the feature collection declaration in the message - the BUFR instance.
BUFR simple features are continuous coverages, discrete coverages (code flags) and operation declarations on the simple features (data types) which modify those feature types.

Section 3 iis a set of table references to BUFR Table D (for generic feature collections defined for each Table A BUFR specialisation) Table C for operations (such as replications representing meteorological observations at multiple locations and times) on the section 3 references, and references to Table B (simple features and further operations).

When the Table D references are expanded to sets of Table B references, and the Table C operations are performed on the section 3 elements, the result is an expanded set of simple Table B references which give a one-to-one mapping to the values implicit in Section 4.

The elemental Table B references contain codes for units of measure, numerical offsets and multipliers, and the number of bits in which each value is stored in Section 4. There can also further information on any extra compression used on Section 4 values. When this compression is expanded, the bit sections can be separated and the numerical multipliers and offsets applied to the number represented by the bit string to return fixed-point values for features which are continuous coverages. For example this might be a temperature specified to 0, 1 or 2 decimal places in degrees Kelvin.
For discrete coverages, the Table B entries refer to BUFR code tables describing distinct weather elements, such as present weather codes refering to mist, fog, thunder etc..

The BUFR tables define in full detail every non gridded feature known to WMO. The Tables also allow individual instances to be coded extremely concisely.

Although this seems very different from ISO using GML, the parallels are clear. BUFR Tables are feature catalogues. They are complete descriptions of WMO data and the table mechanism - like ISO feature catalogues - can be extended to many other domains.

Figure 8: the BUFR Message - a data-driven feature

4.9 Figure 9: BUFR descriptor operators

There are a number of types of BUFR operators but they split broadly into model operators and coding operators. These are specialisations of the general class of BUFR_Operators.

Of the coding operators (to the right of the note connector) the dataCompressionOperator is distinct in that it is not represented in the BUFR Tables A, B, C or D and has no F_X_Y index. It is flagged as enabled with the compressionFlag of the attributes of the dataDescriptionSection3 in Figure 8, but it must be recognised as an operation or activity.

The replicationOperator is a single operator class of its own. It is identified by the indicator (1st level index) having the value 1, and the 2nd level index is the number of terms to be replicated; the 3rd level index is the number of replications.

The dataDescriptorQualifierOperators are the operator implementation of the special Table_B class which modifies other Table B features of Figure 5. These are explained in Figure 11.

The coordinateOperator sub-class is the operator implementation in the coding process of the BUFR_table_B_coverage classes of Figure 4. These operators are detailed in Figure 10.

The operatorDescriptor subclasses of BUFR_descriptorOperators are all the classes of Table C. This is shown in Figure 12.

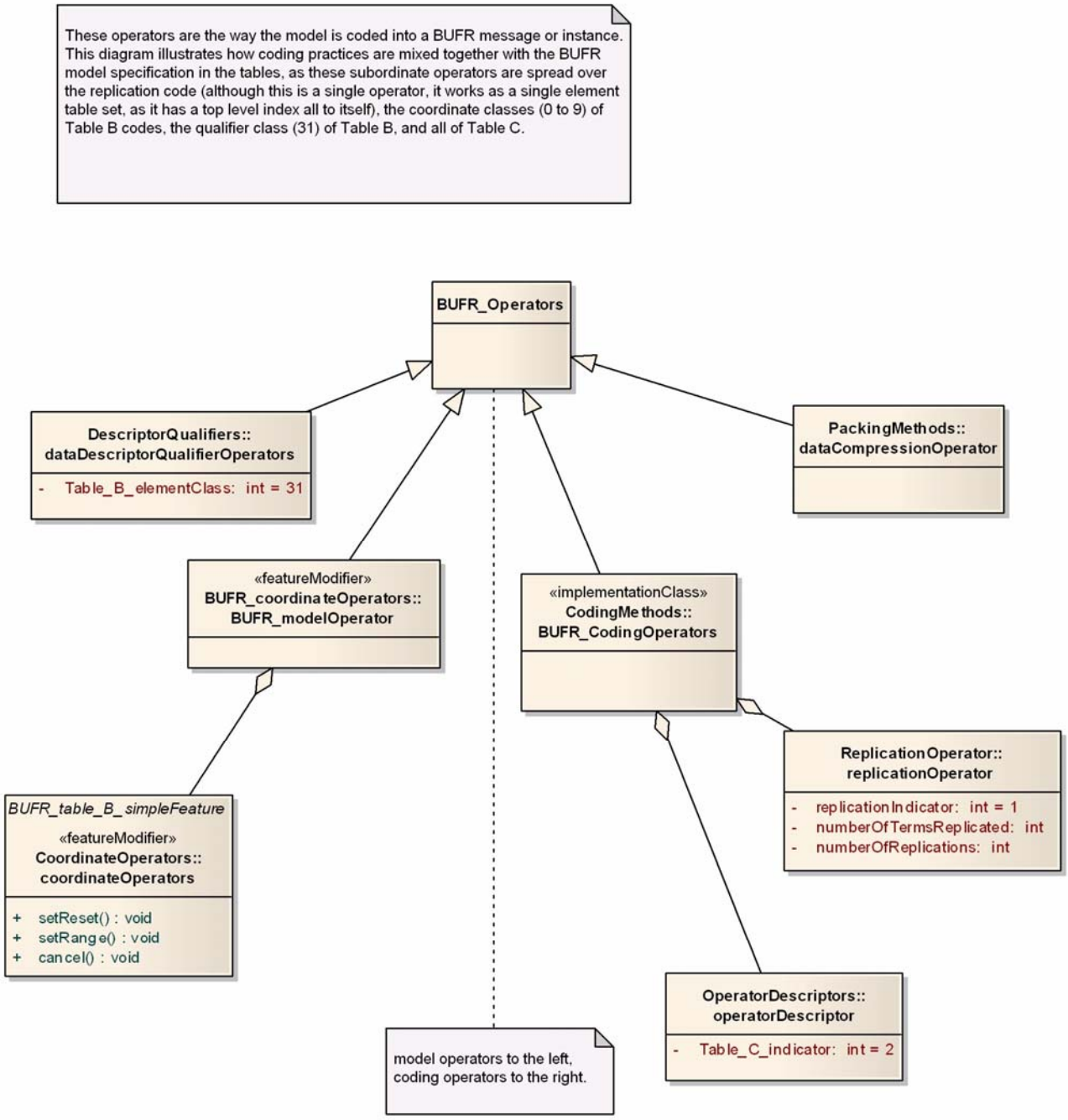


Figure 9: BUFR Operators.
model operators (feature modifiers) and coding operators

4.10 Figure 10: BUFR coordinate operators –coverage specification in an instance

The classes of coordinateOperator are all identified as coverage operators, though the precise function is to group the following terms. This grouping remains in place until it is reset by a following re-definition of the coordinate value. Some of the operators have a cancel function, and others have the option of declaring a range if two operators perform in sequence.

The timeCoverageOperator class can set an absolute time, but also set a time increment, displacement or duration. There are also local displacements which can be set on a restricted set.

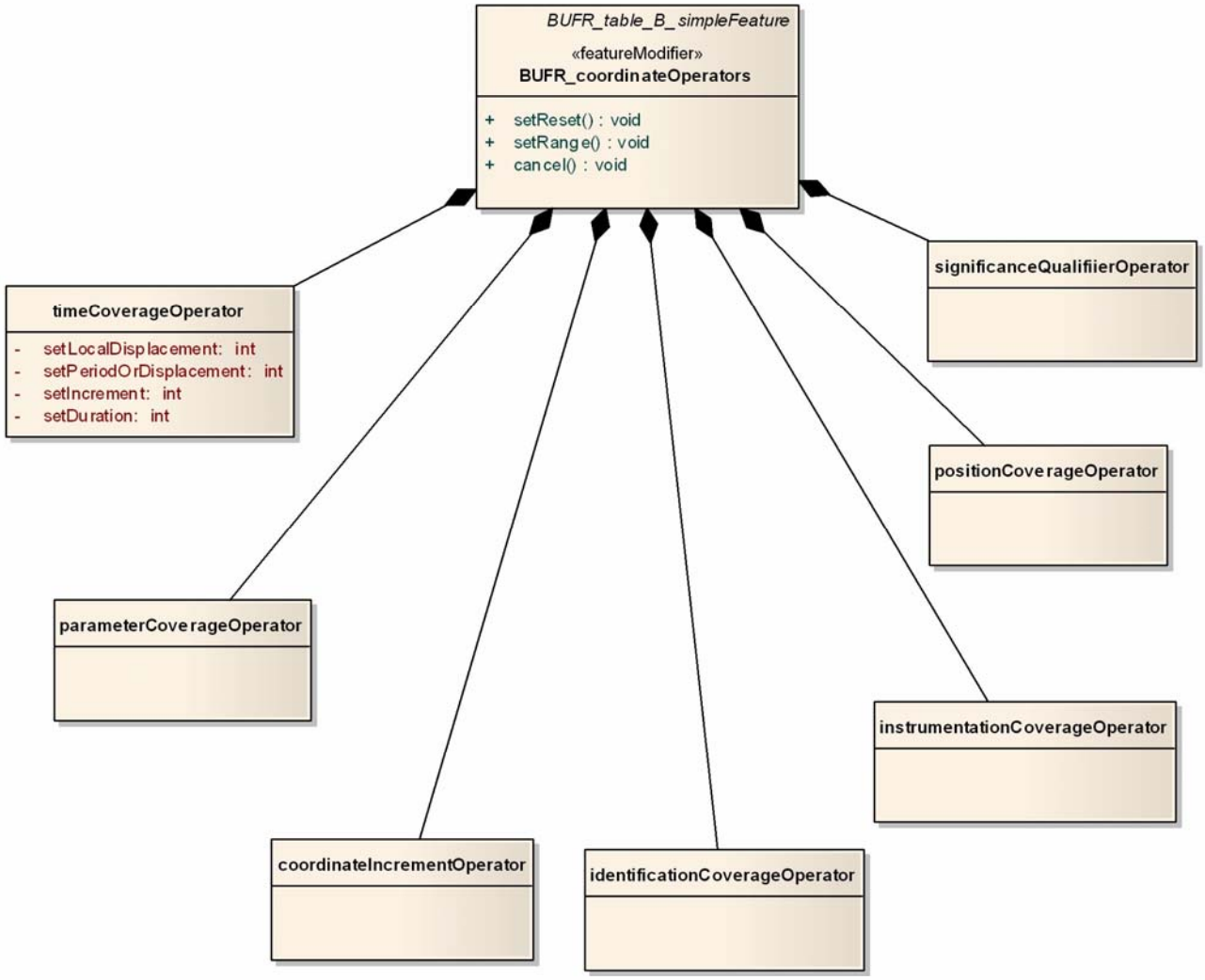


Figure 10: BUFR coordinate operators. grouping, coverage grid and feature attribute specification within a BUFR_Message (feature instance)

4.11 Figure 11: Operators qualifying the data descriptors

Table B class 31 is the set of dataDescriptorQualifierOperators. This is a sophisticated mechanism to allow a BUFR message or instance to contain variable numbers of elements of one type or a set of types. It is frequently used with the coordinate operators to define number of points in the internal coverage grid.

There are 3 component classes of operator.

The “delayed” mechanism delayedReplicationOperator denotes that the collections or templates do not have a predefined number of terms, so a variable length sounding[12], or an array of values may follow. While the replication operator modified the descriptor or element numbers, the data sequence will follow the actual number defined in the instance. The delayedRepetitionOperator on the other hand, allows for repeated values of the data equivalent to a run-length packing mechanism.

The dataMissingOperator allows for declaring the data to be present or missing using a single bit in the data value stream, as an alternative to using the full bit length of the element to give the data missing value assuming it is present in the definition of the feature.

[12] A “sounding” is a profile in the vertical of the atmosphere or ocean. Measured or estimated values of different parameters (e.g. temperature, wind, pressure, density, current) are taken at different points in the vertical profile.

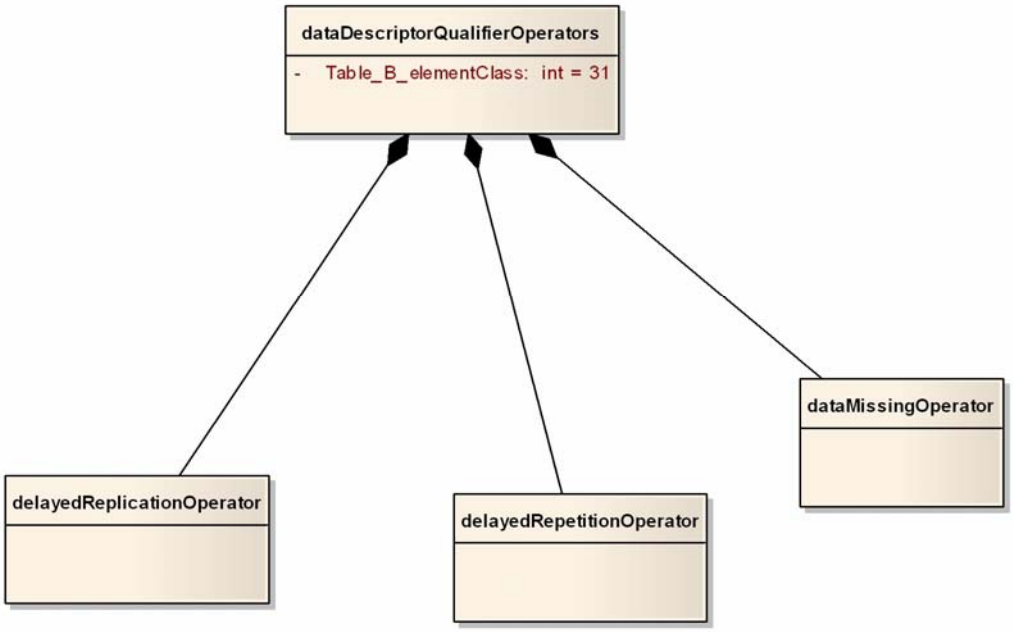


Table B class 31 is the set of data descriptor qualifier operators. These are a sophisticated mechanism to allow a BUFR message or instance to contain variable numbers of elements of one type or a set of types. It is frequently used with the coordinate operators to define number of points in the internal coverage grid. The "delayed" mechanism denotes that the collections or templates do not have a predefined number of terms, so a variable length sounding, or an array of values may follow. While the replication operator modified the descriptor or element numbers, the data sequence will follow the actual number defined in the instance. The delated repetition operator on the other hand, allows for repeated values of the data as a run-length packing mechanism.
The data missing operator allows for declaring the data to be present or missing using a single bit in the data value stream, as an alternative to using the full bit lebgth of the element to give the data missing value (if it is defined).

Figure 11: Operators qualifying the data descriptors

4.12 Figure 12: Coding operators: modifying the attributes or adding annotation.

This demonstrates the class of operatorDescriptors which includes all the Table C descriptors.

The biggest class, and easily the most used of Table C operators are the dataAttributeOperators. The fixed attributes of Table B features explained in figure 6 can be dynamically modified by these operators. The class is composed of the unitsChangeOperator (which modified units-of-measure), the referenceChangeOperator, scaleChangeOperator and the dataWidthOperator. These can be set to apply widely or reset to revert to Table B values.

All these operators have effects entirely on the coding formats. The remaining Table C operators are coding operators, but they add annotation, associated fields or quality assessments which may have persistent effects on the coding. However, these are mostly used for quality assessment, data monitoring and verification, and not to the mainstream Table D sequences.

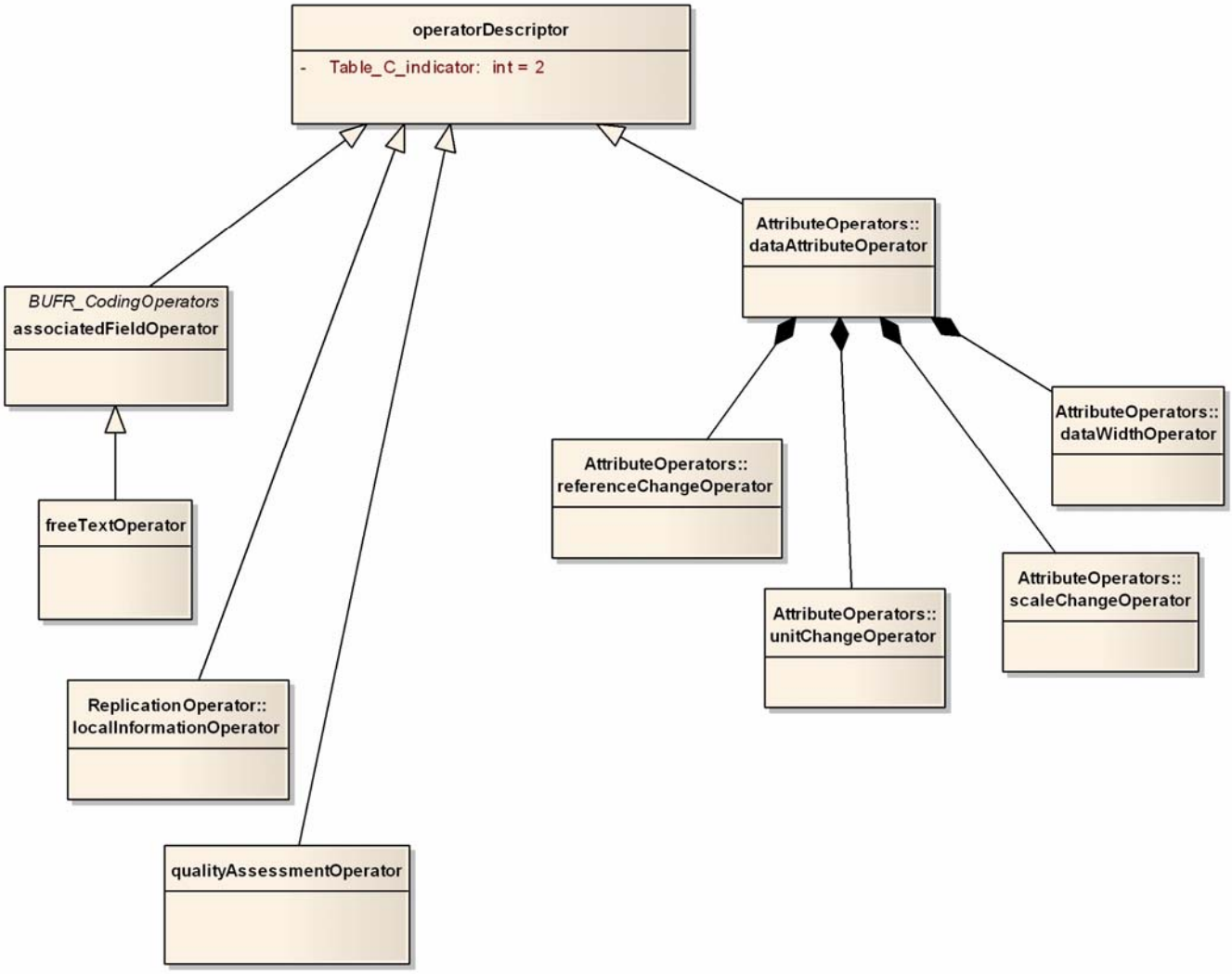


Figure 12: Coding operators modifying the attributes or adding annotation

5 Route-map to recast the BUFR model and to derive a GML application schema

5.1 GML Application schemas

The Open Geographic Consortium[13] (OGC) developed Geography Markup Language[14] (GML) as a language (an XML grammar) to specify geographic application schemas. GML is both a modelling language for geographic systems describing geographic "features" and an exchange format for geographic data on the Internet. The use of "features" and "coverages" in the Atmospheric Sciences domain is described in section 4.2.

The original GML model was based on the W3C's (World Wide Web Consortium[15]) Resource Description Framework (RDF), but later OGC developed XML schemas within the GML framework. Some RDF properties, such as child elements retaining properties of the parent objects, are quite difficult to convert to the deliberately non-hierarchical BUFR model.

An application schema is a domain- or community-specific specification, which supports data interoperability in a geographic model within and across communities. GML has been adopted as an ISO standard (ISO 19136) and ISO 19103 defines rules to design UML through which to develop the GML application following the rules given in Annex E of ISO 19136.

OGC also have an Abstract Specification, the "Observations and Measurements" Schema[16] which defines an abstract model and an XML schema encoding for observations and measurements. This is expanded with all relevant ISO 191xx schemas in a template to create GML Application Schemas in the "Hollow World"[17] of Rob Atkinson and Simon Cox.

5.2 BUFR in GML

To develop BUFR features in GML, we need to create an XML schema representing the BUFR elements. However the large scale of the BUFR model, and the data-driven nature of the schemas contained in the BUFR_Message, make a full representation of the BUFR model inconceivable.

So the question arises – "**Why might we wish to express BUFR in GML?**", when BUFR is an older, operational standard which has such a large data model, compared to GML or ISO 191xx standards.

The follow points are worthy of comment:

---

[13] OGC http://www.opengeospatial.org/ogc
[14] GML http://www.opengeospatial.org/standards/gml
[15] W3C http://www.w3.org/ and RDF http://www.w3.org/RDF/
[16] O&M Simon Cox:
http://portal.opengeospatial.org/modules/admin/license_agreement.php?suppressHeaders=0&access_license_id=3&target=http://portal.opengeospatial.org/files/index.php?artifact_id=22466
[17] https://www.seegrid.csiro.au/twiki/bin/view/AppSchemas/HollowWorld

- BUFR is a highly developed format for specialists. GML Application Schemas are intended for cross-community interoperability.
- BUFR software is freely available from several sites, but it is in a form most suitable for specialist WMO use.
- GML is based on XML and is adopted as a standard under ISO 19136. While tools and services are still being developed, there will be many offerings, using both proprietary and free software, capable of reading, displaying and using GML based data on universal PCs.
- There are many external requirements forcing a BUFR-GML conversion. Some are:
    - WMO's WIS programme is committed to using modern Internet tools. WIS will use WMO's Core Profile of the ISO 19115 Metadata standard as integral to WIS data exchange.
    - WMO and ISO have agreed to adopt each others' standards and WMO is looking to express its data standards, developed over decades, in a unified Conceptual Data Model.
    - Under the European Directive of the Single European Sky (SES[18]), EUROCONTROL are committed to develop ISO 191xx based data models for Aviation OPMET data, which are subsidiary classes within BUFR. This is needed to work with other aviation data under the AIM[19] (Aeronautical Information Management) data model. This has a GML formulation in those parts which are geography based.
    - The European INSPIRE Directive requires data and services to be modelled in UML and expressed in a GML-based format. While it is intended that this does not refer to all meteorological data, that data which non-specialist users will see, will be covered.
- BUFR data is heavily compressed while GML versions of the same data will have considerable bloat. This means that not all BUFR data can be or needs to be converted to GML. This suggests that an intermediate solution is required, with a clear route to and from both BUFR and GML.
- As discussed in section 3.1 and this section, BUFR is a dynamic code form, where new features can be defined within a BUFR message. The combinatoric explosion of potential feature definitions means that a complete listing of feature types (as in a feature catalogue) in inconceivable. We need a process to get to GML with perhaps only a common sample of features actually listed in GML.

So, there are strong reasons for having a way of converting BUFR Features into a Feature Catalogue, and being able to express them in a GML Application Schema. There are, however also strong reasons for not doing this with the whole BUFR model.

The following sections outline a process for converting BUFR into an XML dialect, then being able to create components of a Feature Catalogue from the feature types represented there. This allows sections of the BUFR model to be created as Feature Catalogues in separate analyses, but gives a mechanism to ensure they are compatible.

---

[18] http://www.eurocontrol.int/ses/public/standard_page/sk_ses.html

[19] *From AIS to AIM, A Strategic Road Map for Global Change, version4*, Eurocontrol, 2006, http://www.eurocontrol.int/aim/public/standard_page/aim_library_intro.html#strategy

5.1 Figure 13: Conversion of a BUFR message to XML

This describes the process involved in converting any BUFR message or instance to XML. By extension we can also derive a schema for this XML by imposing controls on the schema, and in a very similar way, the feature types defined in the BUFR message, can be expressed as components of a feature catalogue.

That the BUFR Message can be converted to an XML version is because the BUFR message is a fully structured form where the meaning and content are resolved by reference to the BUFR tables and application of the coding operators to decode the message.

The metadata, data description section and data sections contain all the message information. The decoding operation is applied next (this is expanded in figure 14, section 5.2) and from this the expanded sequences of BUFR descriptors and the expanded data sequence are produced. This is entirely the normal process in decoding a BUFR message, with a few extra additions to retain the BUFR sequence hierarchy.

At this point this is an instance of a full markup language. The two files give a one-to-one mapping from BUFR descriptor identifiers to the data values.

The second step is the conversion to an XML instance. Since the expanded sequence files are already a fully structured dataset, there is considerable freedom in deciding the form and dialect of the XML.

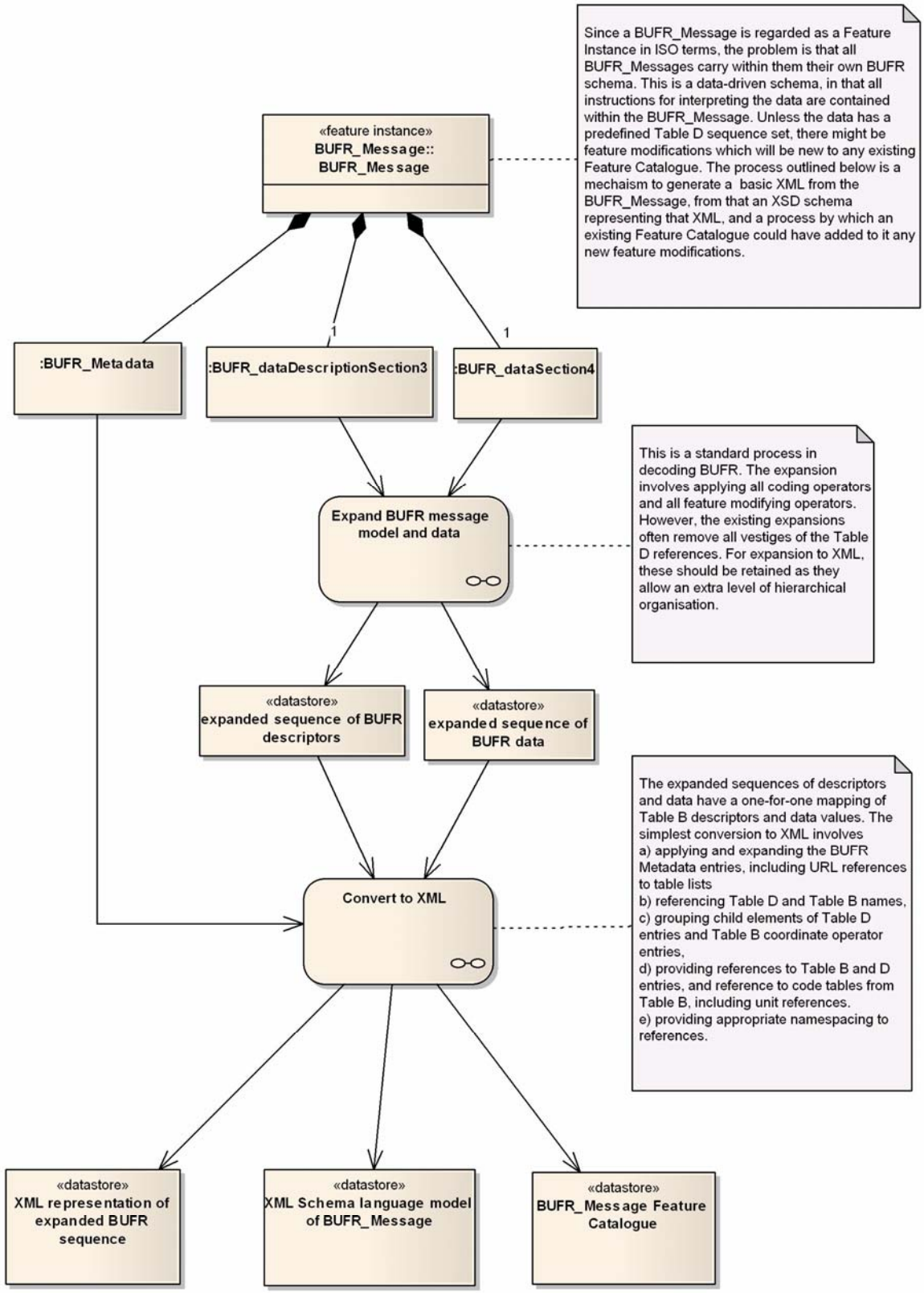


Figure 13: Action Diagram describing conversion of BUFR_Message (feature instance) to basic XML, and additions to a feature catalogue

5.2 Figure 14: Expanding/decoding a BUFR Message/Instance

This is a version of the normal decoding process of a BUFR message.

There are a number of detailed process decisions which will be made here, but they are not included in the diagram.

The first stage expands all the Table D sequences to include all Table B References. In some BUFR decoders, the Table D references are removed at this point reducing the hierarchy of the descriptor file to a one sibling level. It is preferable not to lose this information, and ideally the Table D tags should be retained, and end tags should be introduced to delimit the Table D sequences.

The logical holding of the descriptor file at this stage is a sequence of BUFR identifiers: Table D identifiers wrapping Table B identifiers, and Table C operator identifiers, replication identifiers etc.. The identifiers still reference the table entries, and all other information can still be accessed through the tables. Whether that is expanded into the actual rather than logical datastore is an implementation decision.

The next stage is to retrieve the Table B information. The BUFR data expression information (scaling, units, reference value and data width) are all that is needed for the logical process: more can of course be added to the actual datastore.

The multiplicity, the number of data subsets in the message (numberOfSubsets in the BUFR_dataDecriptionSection3 of figure 8) is needed here. Much of the compression in a BUFR message is gained by storing the number of times in which the base set of descriptors is repeated in an instance. There are other operators which refer to the data and clever ways to log missing data which mean that repeated descriptor sets are not necessarily identical. The logical datastore at this point has all Table D sequences and Table B references for one subset, followed by the same sequences and references in the second and subsequent subsets.

The following stage checks whether compression has been enabled. If it has then the data section is not in the same sequence as the descriptor datastore, but is a transposed array, with all values for the first descriptor, then all values of the second descriptor, and so on. The extra compression applies to each descriptor type in turn. Decompressing involves expanding the compression and transposing the data array back to match the descriptor sequence.

The final stage follows the decompression operation, if it is applicable. Here any Table C operators which temporary modifications to the Table B data specifications are applied and fixed replication, delayed repetition or replication operations are applied to the data store (Figures 11 and 12).

The outputs are the expanded descriptor set, and expanded dataset, such that the Table B descriptors are one-to-one mapped with the content values. All coding operators are complete and are no longer listed in the descriptor logical datastore.

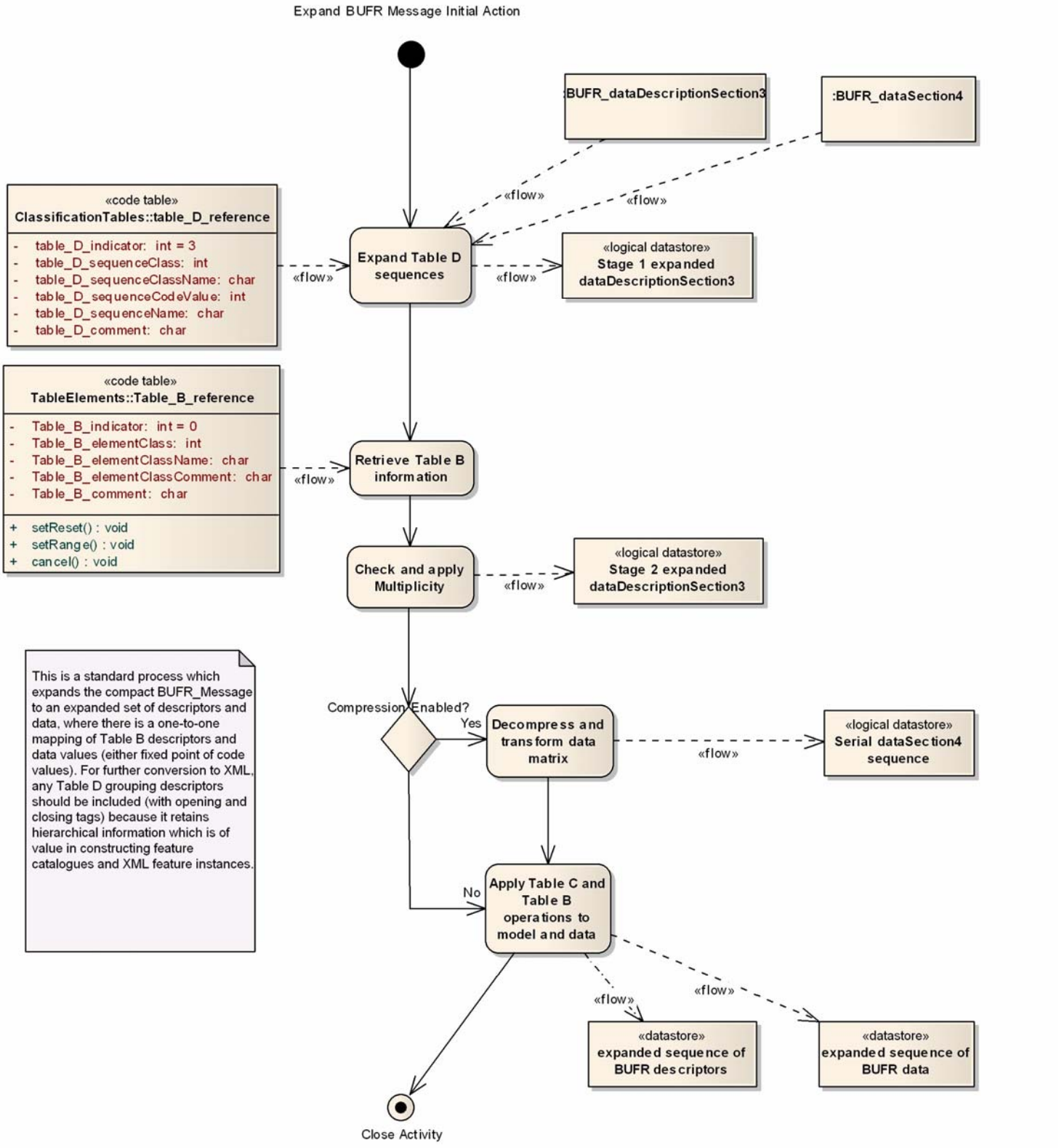


Figure 14: Action Diagram depicting the operations needed for the common task of expanding a compact BUFR_Message (feature instance).

5.3 Figure 15: Conversion to XML

With a fully structure pair of extended BUFR descriptor and data datasets, and with the metadata from the BUFR message, a fully defined XML message is just a transformation.

However, to get an exchangeable XML, and even more to get XML based on a GML application Schema requires a lot of decisions which this paper will only describe.

5.3.1 BUFR Tables in XML form

In the process described in Figure 15, the BUFR Table references still have to be resolved. Either the information has to be included in the XML message, or a reference to it needs to be encoded in XML. If an XML reference is included, this means that the BUFR tables have to be presented in a form which individual element references can be accessed via URLs.

This requires the BUFR tables themselves to be published in an XML form. There have been a number of versions of the BUFR tables created in XML form, but for this to be a proper WMO supported process, this has to be developed with the support of the WMO Expert Team which maintains the BUFR tables.

In doing so, the XML can be augmented in several ways. Each Table D entry and each Table B entry has to have a formal XML tag name. Each sub-class of Table D and Table B has to have its own namespace. These names will of necessity be specialised names, reflecting the scale of the BUFR model. These specialised names should be closely related to the BUFR descriptions in each Table B row.

However the specialised BUFR names may also need to have aliases which are common names. These common names perhaps should be linked to existing ontologies[20] and other definitions[21], such that there is an agreed mapping from WMO specialised BUFR names to names used more widely. A second reason for this is that there are often many possible BUFR elements which might apply which would confuse the non specialist (this is really unwanted detail rather than excessive complexity).

As an example, a non specialist would have difficulty matching a request for temperature to the BUFR descriptors. It is very difficult to decide which of the many entries of the BUFR Temperature class 12 are wanted, never mind the many possible vertical or significance coordinate modifiers applied to the simple features. That the first match to “air temperature” is the “temperature/dry bulb temperature” (0-12-001) is serendipitous, but there are other good matches (0-12-004, 0-12-101, 0-12-104 without specifying coordinate operators).

---

[20] NASA Semantic Web for Earth and Environmental Terminology (SWEET) ontology http://sweet.jpl.nasa.gov/ontology/ .
Marine Metadata Interoperability Project ontologies
http://marinemetadata.org/examples/mmihostedwork/ontologieswork

[21] Global Change Master Directory http://gcmd.nasa.gov/
Climate and Forecast Metadata Convention http://www.unidata.ucar.edu/software/netcdf/conventions.html

5.3.2 XML Conversion

Figure 15 gives the first activity as initialising the XML. The BUFR message identification section is translated into appropriate metadata entries, and the references to the originating Centre and sub-centre are expanded via the Common Code Tables. Since these are very static values, another option would be to keep these as references, but redirect the reference to appropriately expanded Citation sections based on ISO 19115.

This initialisation also should fill out namespace references from the WMO tables.

The second action, to expand the Table D and Table B tags would use the tag names to fill the element names (discussed in 5.3.1 above) and the BUFR references would be inserted as XML attributes. Table D elements should be closed at the Table D end tags. The BUFR coordinate operators would use the Table B tag names, but when the coordinate is repeated, updated, closed, or when a sub-section ends, a closing tag would be inserted.

Simple Table B entries would be expanded. For numeric values, the content would be filled and the units added as an attribute. For Code table or Flag table values, the code table references would already be available as the Table B BUFR reference, although it might need a different high-level pointer to the appropriate code table. The “units” attribute might be related by a code table entry. How the value of the code table entry is included in the XML is another implementation decision. The code value could be kept as an attribute, with either no content, or the code description as the content. Since flag tables allow multiple entries and the descriptions themselves can be very large, it may be easier to leave filling the content until a later stage.

The output of this activity is well-formed XML which does not have an XML Schema (unless perhaps, it is a repeat of a previous BUFR message of the same type). As described, this is not an XML instance of a GML Application schema. This point is discussed below in 5.3.3.

A message specific schema can be generated automatically – most XML tools allow well-formed XML to be given a schema – but there are also set rules in BUFR which could be used to make such a schema more specific to BUFR. BUFR types are fully specified and multiplicities can be decided by keeping a record of the fixed and dynamic repetitions in the expanded descriptor logical datastore. For optional elements, in the BUFR Table D sequences the optionality is mostly overridden. The Table D sequences contains the full set of descriptors, even if the instance values are set missing.

Such an XML Schema is not very useful. The BUFR message itself demands that the instance is valid BUFR, so it must be valid XML.

5.3.3 Creating a Feature Catalogue.

While it is not likely that all possible BUFR features could be incorporated into a Feature Catalogue, in practice some subsets of the BUFR model should be. The BUFR Tables, and specifically the Table D sequences represent most of the common feature types in use, but it should be remembered that Table D sequences are not required by the BUFR model, and consequently restricting attention to Table D sequences will not give a universal solution.

The BUFR tables have most of the information to create a BUFR feature catalogue in a vocabulary for specialists. However the real requirement for a feature catalogue under a GML application is from non-specialist users. These users need to refer to a smaller set of common features of atmosphere in a standardised way and to combine this information with other non-meteorological geographic features in, commonly, a map. They need protection from the detail of the BUFR model, the BUFR features and the specialist vocabulary.

It is likely that different users of different types of data will need distinct feature catalogues. The problem then is to create that capability while being able to reconcile the catalogues in a consistent way. There are communities of users (aviation, civil engineers, tropical storm forecasters) who need to define a community vocabulary within or associated with the BUFR tables while at the same time excluding unwanted BUFR elements from the vocabulary.

The process listed in the last activity of Figure 15 recognises that the feature types contained in a BUFR instance could be used, with the BUFR tables, to reconstruct those feature types contained in a "nominal" BUFR feature catalogue. While most of the information is already contained in the BUFR tables, there will also be extra information needed which may be added to the BUFR table in the form of translation rules.

5.3.4 Converting to a GML application instance.

The dialect of XML described as a direct transformation of a BUFR message or instance, requires some extra information in the form of translation rules, in order for it to be rendered as a GML application instance.

- The first sort of rule is to allow multiple representations in BUFR to be recognised as essentially the same feature. (see 5.3.1)
- The second kind of rule is to allow common names or aliases.
- A more difficult type of rule is how to interpret BUFR generalised co-ordinate mechanisms to be rendered in a GML instance.

The coordinate mechanism allows 3 different XML/GML mechanisms to be implemented in BUFR. The first two, a grouping mechanism and a coverage grid mechanism are relatively straightforward, and rendering the various representations of record-domain-range within GML coverages should be straightforward. However, the coordinate mechanism, particularly in the significance qualifiers of section 4.4.4 also model properties many of which will be recognised as feature attributes. Since feature attributes are child properties of a feature, distinguishing which significance qualifiers are coverage grid measures and which are feature attributes may not be an automatic option: it may require external judgement. In BUFR, all simple features (section 4.5) are children of the coordinate mechanism.

It is likely that several BUFR feature catalogues will need to be created for particular user requirements. This process shows that it will be possible to coordinate different catalogues for different purposes as long as they provide mappings back to the original BUFR tables. This should be treated as an augmentation of the BUFR tales and maintained accordingly by WMO.

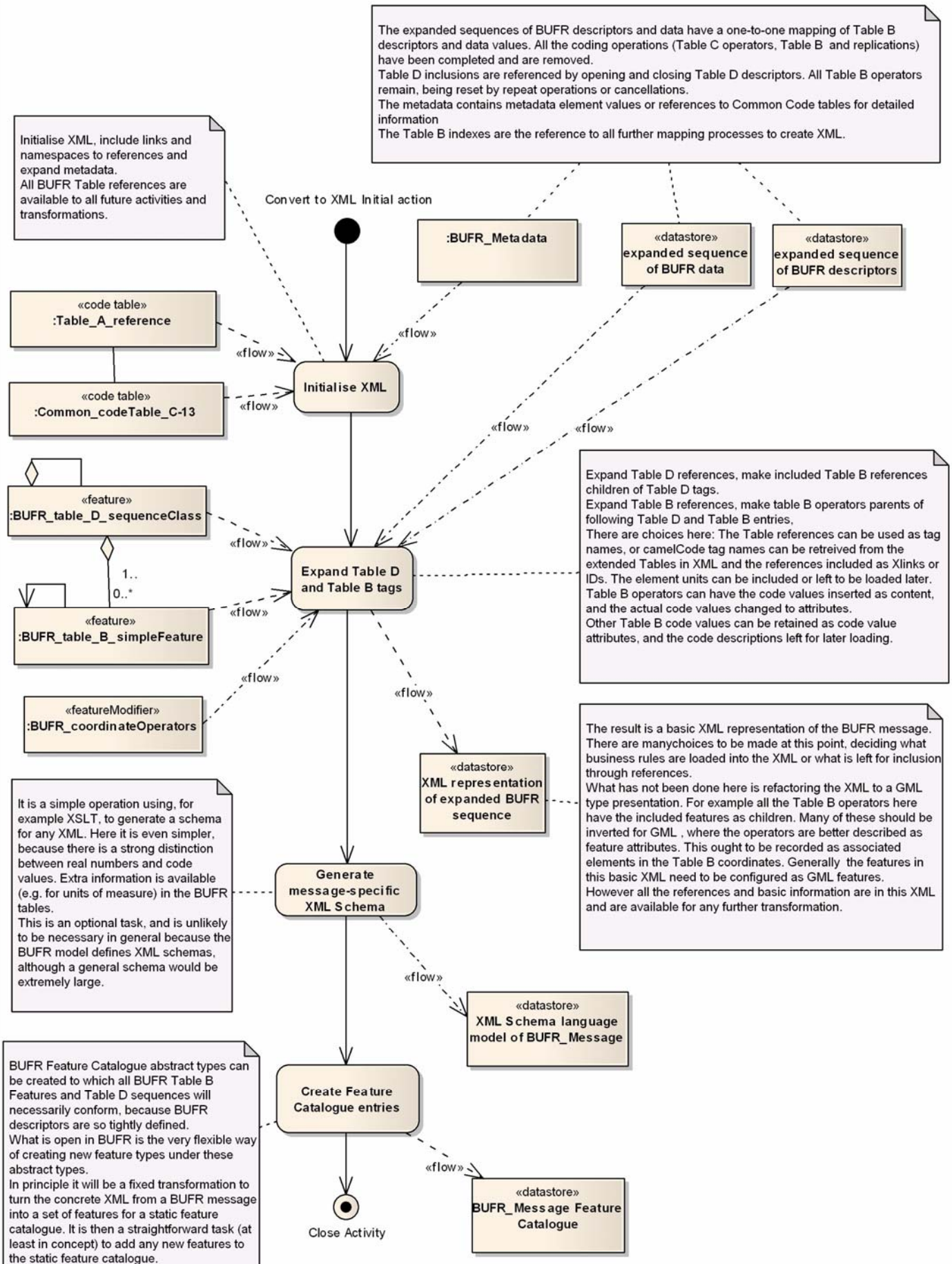


Figure 15: Converting an expanded BUFR descriptor/data instance to XML and elements for a feature catalogue.

6 Conclusions and Recommendations

The BUFR model is a very complete system first made operational in 1988. Since it is so much older than the ISO_TC211 geographic model, it is not defined in the same way. Nevertheless as a complete self-describing system, there are analogous mechanisms for almost all of the ISO concepts.

However, BUFR uses a data-driven model, and the separation of model and instance (feature catalogue and feature instance) in ISO_TC211 is broken: a BUFR message/instance holds it own schema internally, and in a limited way, the schema can define new types of features.

But BUFR is a complete self-describing system, so recasting a BUFR instance into a BUFR dialect of XML is straightforward in principle. In practice it requires some work in order for this to be standardised, and this work should be adopted and maintained as part of the BUFR maintenance process in WMO.

1. Although BUFR tables are published as MS word tables, they should be converted into a standard XML form, where additional properties can be added.
2. These additional properties include
    a. element (descriptor) names, table and sub-table namespaces
    b. common name aliases, including multilingual names
    c. links to common names in other vocabularies
3. All the tables should be linked by XML linking mechanisms, and the XML published in an WMO registry.
4. A BUFR decoder which converts BUFR messages/instances into a BUFR XML dialect should be developed, and the rules to create the dialect should be formalised and published.
5. A first BUFR feature catalogue compatible with ISO 19110 should be created for a nominal user requirement. This should be used to define the translation rules from the BUFR XML dialect for individual feature types.
6. This feature catalogue should be transformed via the “Hollow World” model (footnote 14 section 5.1) to create a GML application schema.

This further work will give form and experience in matching the WMO BUFR (and other) standard(s) with ISO_TC211 standards. While BUFR is a full Data Model in its own right there needs to be a standard way of mapping it to other data models. This should give the basis for a WMO Conceptual Data Model in the modern idiom.

7 References and Code Tables

The first set of hyperlink references are taken from the WMO WWW pages.

7.1 Guide to WMO Table-Driven Code Forms: FM 94 BUFR and FM 95 CREX

Understanding BUFR and CREX.
Layers 1, 2 and 3 *(Note: Layer 3 in English only)* English French Spanish Russian

7.2 WMO Table-Driven Operational Codes –

http://www.wmo.int/pages/prog/www/WMOCodes/OperationalCodes.html

	Operational Codes	Operational on 7 November 2007
BUFR	**BUFR Code Form and Regulations**	Word pdf
	Table A – BUFR. Data Category	Word pdf
	Table B – BUFR/CREX. Classification and definition of data elements	Word pdf
	Definition of Code and Flag Tables associated with Table B – BUFR/CREX	Word pdf
	Table C – BUFR. Data Description Operators	Word pdf
	Table D – BUFR. List of common sequences	Word pdf
CREX	**CREX Code Form and Regulations**	Word pdf
	Table A – CREX. Data Category	Word pdf
	Table B – CREX. Classification and definition of data elements	Word pdf
	Table C – CREX. Data Description Operators	Word pdf
	Table D – CREX. List of common sequences	Word pdf
	CREX Template Examples	Word pdf
COMMON FEATURES:		
	Common Code Tables to BUFR, CREX, GRIB 2 and TAC (Binary and Alphanumeric Codes)	Word pdf
	BUFR/CREX Template Examples, and regulations Templates webpage	
GRIB	FM 92 – **GRIB** Edition 2	Word pdf
	GRIB webpage (to see for GRIB Edition1)	

7.3 TACs Traditional Alphanumeric Codes

For all Traditional Alphanumeric Codes see Manual on Codes

7.4 UML Unified Modelling language

For the formal UML definition see Object Management Group http://www.uml.org/.

7.5 ISO 19100 series of standards

A WMO document describing the ISO 19100 series of standards is available at:
http://www.wmo.int/pages/prog/www/ISS/Meetings/ITT-FWIS_Geneva2004/5(2)_ISO.doc
This document has links to the ISOTC211 at http://www.isotc211.org/outreach/Overview/.

8 Glossary

AIM	Aeronautical Information Management
BUFR	FM 94 BUFR – Binary Universal Form for Representation
CF	Climate and Forecasting convention
CREX	FM 95 CREX – Character form for the Representation and Exchange
CRS	Coordinate Reference System
DSL (or DSM)	Domain Specific Language (or Model)
DTD	Document Type Definition
EUROCONTROL	European Organisation for the safety of air navigation
ET DR&C	Expert Team on Data Representation and Codes
GCMD	Global Change Master Directory
GF-3	General Format 3 (an IOC precursor to BUFR)
GML	Geographic Markup Language
GRIB	FM 92 GRIB – GRIdded Binary
IOC	Intergovernmental Oceanographic Commission
ISO	Not an acronym, but the name of the International Standards Organisation
JCOMM	Joint IOC-WMO technical Commission on Oceanography and Marine Meteorology
METAR	Aviation Routine METeorological report
OGC	Open Geographic Consortium
OMG	Object Management Group
OPMET	Operational METeorological data for Aviation
RDF	Resource Description Framework
SES	Single European Sky Directive
SGML	Standard Generalized Markup Language
SWEET	Semantic Web for Earth and Environmental Terminology (SWEET) ontology
TAC	Traditional Alphanumeric Codes
TAF	Terminal Airfield Forecast
TC211	ISO Technical Commission 211 on Geographic standards.
TDCF	Table Driven Code Forms
UML	Unified Modelling Language
VFR	Visual Flying Rules
W3C	World Wide Web Consortium
WMO	World Meteorological Organisation
WIS	WMO Information System
XML	eXtensible Markup Language
XSD	XML Schema Definition

Annex 1 UML Connections and Relationships

The following has been adapted from ISO documentation. A full description of the ISO conceptual model is in ISO 19103. The formal definition of UML is in the Object Management Group http://www.uml.org/ .

A.1 Associations

An association is used to describe a relationship between two or more classes. UML defines three different types of relationships, called **association**, **aggregation** and **composition**. The three types have different semantics. An ordinary association shall be used to represent a general relationship between two classes. The aggregation and composition associations shall be used to create part-whole relationships between two classes.

The **direction** of an association must be specified. If the direction is not specified, it is assumed to be a two-way association. If one-way associations are intended, the direction of the association can be marked by an arrow at the end of the line.

An **aggregation** association is a relationship between two classes in which one of the classes plays the role of container and the other plays the role of a containee. A **composition** association is a strong aggregation. In a composition association, if a container object is deleted, then all of its containee objects are deleted as well. The composition association shall be used when the objects representing the parts of a container object cannot exist without the container object.

A.2 Generalization

A **generalization** is a relationship between a superclass and the subclasses that may be substituted for it. The superclass is the generalized class, while the subclasses are specified classes. Another common name for a generalisation is an **inheritance**.

A.3 Instantiation/Dependency

A **dependency** relationship shows that the client class depends on the supplier class/interface to provide certain services, such as:

- Client class accesses a value (constant or variable) defined in the supplier class/interface;
- Operations of the client class invoke operations of the supplier class/interface;
- Operations of the client class have signatures whose return class or arguments are instances of the supplier class/interface.

A **realisation** is an instantiated relationship representing the act of substituting actual values for the parameters of a parameterized class or parameterized class utility to create a specialized version of the more general item.

A.4 Roles

If an association is navigable in a particular direction, the model shall supply a “role name” that is appropriate for the role of the target object in relation to the source object. Thus in a two-way association, two role names will be supplied. Different sections of Figure A.1 represent how role names and cardinalities are expressed in UML diagrams.

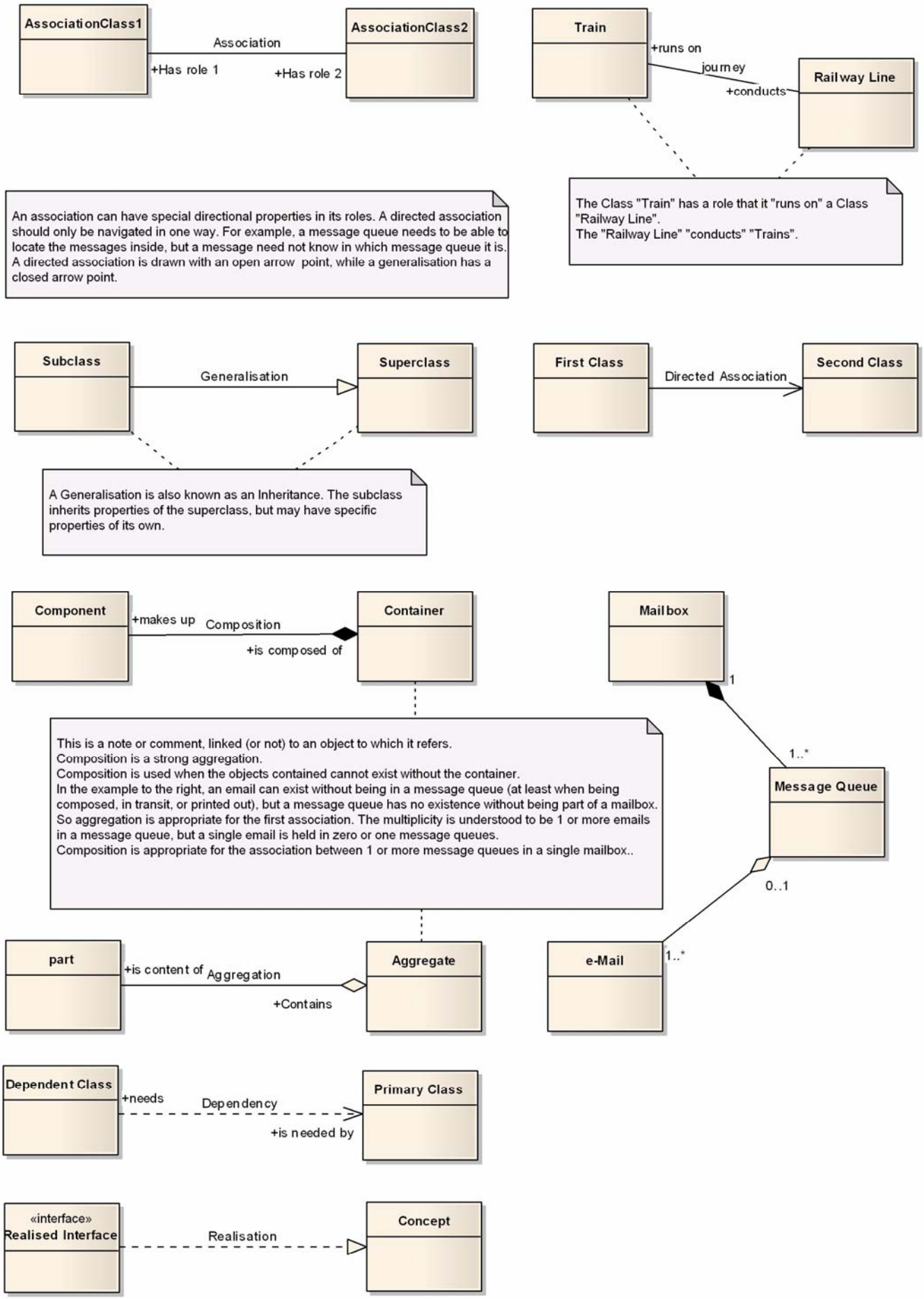


Figure A.1: UML class diagram connections