Draft CF data model

Version 0.3

This document outlines an abstract model for data and metadata corresponding to the CF metadata standard (version 1.5). CF is a primarily a convention for storing data in netCDF, and up to now has not presented a data model. However, the design of CF implies a data model to some extent, and this document is proposed to make it explicit. If adopted as an element of CF, the CF data model description will be updated in line with CF standard. The data model avoids prescribing more than is needed for interpreting CF as it stands, in order to avoid inconsistency with future developments of CF. This document is illustrated by the accompanying UML diagram of the data model.

As well as describing the CF data model, this document also comments on how it is implemented in netCDF. Since the CF data model could be implemented in file formats other than netCDF, it would be logically better to put the information about CF-netCDF in a separate document, but when introducing the data model for the first time, we feel that this document would be harder to understand if it omitted reference to the netCDF information. We propose that these functions should be separated in a later version of the data model. Some parts of the CF standard arise specifically from the requirements or restrictions of the netCDF file format, or are concerned with efficient ways of storing data on disk; these parts are not logically part of the data model and are only briefly mentioned in this document.

In this document, we use the word "construct" because we feel it to be a more language-neutral term than "object" or "structure". The constructs of this data model might correspond to objects in an OO language.

Field construct

The central concept of the data model is a **field construct**. In a dataset contained in a single netCDF file, each data variable usually corresponds to a field construct, but a field construct might be a combination of several data variables. In a dataset comprising several netCDF files, a field construct may span data variables in more than one file, for instance from different ranges of a time coordinate (to be introduced by Gridspec in CF version 1.7). Rules for aggregating data variables from one or several files into a single field construct are needed but are not defined by CF version 1.5; such rules are regarded as the concern of data processing software.

This data model makes a central assumption that each field construct is independent. Data variables stored in CF-netCDF files are often not independent, because they may share coordinate variables, and global attributes apply across all variables. However, this sharing of elements within files can be considered a practical means of saving disk space, and once the information is extracted, we assume that software will be able to alter any field construct in memory without affecting other field constructs. For instance, if the coordinates of one field construct are modified, it will not affect any other field construct. Explicit tests of equality will be required to establish whether two data variables have the same coordinates. Such tests are necessary in general if CF is applied to a dataset comprising more than one file, because different variables may then reside in different files, with their own coordinate variables.

Each field construct may have

- An ordered list of one or more **dimension constructs** (or "dimensions" for short).

Karl Taylor 6/23/12 8:04 AM
Deleted: we view

Karl Taylor 6/23/12 8:05 AM
Deleted: solely as a

- A **data array** whose shape is determined by the dimensions in the order listed, excluding any dimensions of size one. If there are no dimensions of greater size than one, the data array is a scalar. Dimensions of size one are omitted because their position in the order of dimensions makes no difference to the order of data elements in the array. The elements of the data array must all be of the same data type, which may be numeric, character or string.
- An unordered collection of **auxiliary coordinate constructs**.
- An unordered collection of **cell measure constructs**.
- A **cell methods construct**, which refers to the dimensions (but not their sizes).
- An unordered collection of **transform constructs**.
- Other **properties**, which are metadata that do not refer to the dimensions, and serve to describe the data the field contains. Properties may be of any data type (numeric, character or string) and can be scalars or arrays. They are attributes in the netCDF file, but we use the term "property" instead because not all CF-netCDF attributes are properties in this sense.
- A list of **ancillary fields**. This corresponds to the CF-netCDF `ancillary_variables` attribute, which identifies other fields that provide metadata.

All the components of the field construct are optional. The data array would be missing if the field construct serves only to define a coordinate system, which we call a **space**.

The CF-netCDF `formula_terms` (see also **Transform constructs**) and `ancillary_variables` attributes make links between field constructs. These links are fragile. If a field construct is written to a file, it is not required that any other field constructs to which it is linked are also written to the file. If an operation alters one field construct in a way which could invalidate a relationship with another field construct, the link should be broken. The user of software will have to be aware of these relationships and perhaps generate new ones when they might be applicable and useful.

Dimension construct

A dimension construct must contain

- A **size** (an integer greater than zero), which can be equal to one. In CF-netCDF, there is a formal distinction between scalar coordinate variables and size-one coordinate variables, but they are logically the same; CF-netCDF supports scalar coordinate variables for simplicity and convenience in the netCDF file. An example of a size-one dimension is a vertical dimension for 1.5 m height (the height where surface air temperature is commonly measured). In this data model, a CF-netCDF scalar coordinate variable is regarded as a dimension construct with a size of unity.

A dimension construct may also optionally contain

- A one-dimensional numerical **coordinate array** of the size specified for the dimension. If the size of the dimension is greater than one, the elements of the coordinate array must all be of the same numeric data type, they must all have different non-missing values, and they must be monotonically increasing or decreasing. Dimension constructs cannot have string-valued coordinates. In this data model, a CF-netCDF string-valued coordinate variable or string-valued scalar coordinate variable is not part of the dimension construct, but instead is accommodated by an auxiliary coordinate construct.
- A two-dimensional **boundary coordinate array**, whose slow-varying (second in Fortran) dimension equals the size specified by the dimension construct, and whose fast-varying dimension is two, indicating the extent of the cell. For climatological time dimensions, the bounds are interpreted in a special way indicated by the cell methods.
- Properties (in the same sense as for the field construct) serving to describe the coordinates.

Karl Taylor 6/23/12 8:09 AM
Comment [1]: What's the difference? Should you distinguish among the various numeric types (float, integer, binary, etc.)?

Karl Taylor 6/23/12 8:47 AM
Comment [2]: I don't think this last sentence adds anything, and it kind of confused me, since, in fact some field constructs won't have any data in that space.

Karl Taylor 6/23/12 8:45 AM
Deleted: Thus a field construct can be regarded as a space with data in that space.

Karl Taylor 6/23/12 8:18 AM
Comment [3]: This is confusing to me. Why should the data model care how the information is stored in files. Conceivably, couldn't a construct be split across several files? Perhaps this sentence should be dropped or make reference to the CF-netCDF requirements that go beyond the data model.

Karl Taylor 6/23/12 8:19 AM
Deleted: remake

Karl Taylor 6/23/12 8:19 AM
Deleted: them if

Karl Taylor 6/23/12 8:59 AM
Deleted: It

Karl Taylor 6/23/12 8:21 AM
Deleted: d

Karl Taylor 6/23/12 8:30 AM
Comment [4]: The last part of this sentence made no sense to me, so I simply deleted it.

Karl Taylor 6/23/12 8:25 AM
Deleted: corresponds to

Karl Taylor 6/23/12 8:25 AM
Deleted: (not a dimension construct),

Karl Taylor 6/23/12 8:29 AM
Deleted: with a dimension whose coordinate construct has no coordinate array

Karl Taylor 6/23/12 8:29 AM
Deleted: .

In this data model we permit a dimension not to have a coordinate array if there is no appropriate numeric monotonic coordinate. That is the case for a dimension that runs over ocean basins or area types, for example, or for a dimension that indexes timeseries at scattered points. Such dimensions do not correspond to a continuous physical quantity. (They will be called **index dimensions** in CF version 1.6.)

Auxiliary coordinate construct

An auxiliary coordinate construct can be used to help define a space and must contain

- A list of some (at least one) of the dimensions of the field construct in any order. Dimensions can be of length 1.
- A coordinate array with dimension sizes corresponding to the list of dimensions of the auxiliary coordinate construct. The elements of the coordinate array must all be of the same data type (numeric, character or string), but they do not have to be distinct or monotonic. Missing values are not allowed (in CF version 1.5).

An auxiliary coordinate construct may also contain

- A boundary coordinate array with all the dimensions, in the same order, as the coordinate array, and a fastest-varying dimension (first dimension in Fortran) equal to the maximum number of vertices needed to describe the cells.
- Properties serving to describe the coordinates.

Auxiliary coordinate constructs correspond to auxiliary coordinate variables named by the `coordinates` attribute of a data variable in a CF-netCDF file. CF recommends that there be auxiliary coordinate constructs of latitude and longitude for horizontally varying data on a non-Cartesian latitude-longitude grid. As for dimension constructs, auxiliary coordinate constructs included in different field constructs are independent in the data model.

Cell measure construct

A cell measure construct may contain

- A list of some or all of the dimensions of the field construct in any order.
- Properties to describe itself.

A cell measure must contain

- A **measure property**, which indicates which metric of the space it supplies e.g. cell areas.
- A **units property** consistent with the measure property e.g. m2.
- A numeric array of metric values having the dimensions listed, excluding any dimensions of size one, or a scalar metric value if no dimensions of size greater than one are given. The elements of the array must all be of the same data type. It is assumed that the metric is only a dependent on the dimensions specified, and for fields that depend on additional dimensions, the metric values are implicitly propagated along those dimensions.

In CF-netCDF files, variables named by the `cell_measures` attribute of the data variable are instances of cell measure constructs. As for dimensions, cell measures constructs of different field constructs are independent in the data model.

Karl Taylor 6/23/12 8:41 AM
Deleted:

Karl Taylor 6/23/12 8:50 AM
Comment [5]: Note that this seems to contradict the phrase I deleted above that said "with a dimension whose coordinate construct has no coordinate array". That's one reason I deleted that earlier phrase.

Karl Taylor 6/23/12 8:51 AM
Deleted: If there is a dimension with size greater than one, t

Karl Taylor 6/23/12 8:52 AM
Comment [6]: What's the difference? See earlier comment.

Karl Taylor 6/23/12 8:55 AM
Deleted: and

Karl Taylor 6/23/12 8:56 AM
Deleted: of each cell.

Karl Taylor 6/23/12 8:35 AM
Deleted: to

Karl Taylor 6/23/12 8:39 AM
Deleted: if there is two-dimensional horizontal variation but the horizontal coordinates are not latitude and longitude.

Karl Taylor 6/23/12 8:40 AM
Deleted: of

Karl Taylor 6/23/12 8:59 AM
Deleted: and

Karl Taylor 6/23/12 9:00 AM
Comment [7]: Don't we need a controlled vocabulary for this?

Karl Taylor 6/23/12 9:02 AM
Deleted: If there is a dimension with size greater than one, t

Karl Taylor 6/23/12 9:04 AM
Deleted: does not depend on any of the dimensions of the field which are not specified,

Karl Taylor 6/23/12 9:05 AM
Deleted: these

Karl Taylor 6/23/12 9:07 AM
Deleted: cell measures constructs correspond to

Cell methods construct

The cell methods construct describes how the data values represent variation of a field within cells. It corresponds to the `cell_methods` attribute of the data variable in CF-netCDF files. It is an ordered list, because the methods specified are not necessarily commutative. Each entry of the list specifies either one or more dimensions, or a CF standard name (to describe variation with respect to a quantity that is not recorded as a dimension of the field), and a method e.g. `mean` (CF Appendix E). Special methods indicate climatological time processing.

Karl Taylor 6/23/12 9:09 AM
Deleted: the quantity

Karl Taylor 6/23/12 9:10 AM
Deleted: which

Transform constructs

A transform construct identifies functions of the coordinates of the field from which auxiliary coordinate constructs can be computed. A transform construct contains

- A **transform name** which indicates the nature of the transformation and implies the formulae to be used. A CF-netCDF file does not explicitly record the formulae; it depends on the application software knowing what to do.
- An unordered collection of **terms**, which are scalar parameters, pointers to dimension or auxiliary coordinate constructs of the field construct, and pointers to other field constructs. Each member of the collection has a particular role in the formulae.

Karl Taylor 6/23/12 9:11 AM
Comment [8]: We will need a controlled vocabulary for this.

Karl Taylor 6/23/12 9:14 AM
Comment [9]: Need controlled vocabulary

Transform constructs accommodate the CF-netCDF attributes `formula_terms`, which describes how to compute a vertical coordinate variable from components (CF Appendix D), and `grid_mapping`, which describes how to transform between longitude-latitude field and the horizontal coordinates of the field construct (CF Appendix F). The transform name is the `standard_name` of a vertical coordinate variable with `formula_terms`, and the `grid_mapping_name` of a `grid_mapping` variable. The scalar parameters are scalar data variables (which should have `units` if dimensional) named by `formula_terms`, and attributes of `grid_mapping` variables (for which the units are specified by the transform construct). The role of each term in the formulae of the transform construct is identified by its keyword in a `formula_terms` attribute, or its attribute name in a `grid_mapping` variable.

Karl Taylor 6/23/12 9:15 AM
Deleted: correspond to the functions of

Karl Taylor 6/23/12 9:17 AM
Comment [10]: In the usual case, don't we use grid_mapping to go from the fields horiz. Coord. To lon/lat rather than the reverse? [I'm not sure about this.]

Other properties

The other properties recognised by this CF data model correspond to attributes listed in CF Appendix A. For field constructs, the allowed properties are `comment`, `history`, `institution`, `long_name`, `references`, `source`, `standard_error_multiplier`, `standard_name`, `title`, `units`. Some of these can be global attributes in a CF-netCDF file. In this data model, it is assumed that any relevant global attribute is also an attribute of every data variable, although in cases where the data variable has an identically named attribute that conflicts with the global attribute, the global attribute does not apply. Each field construct in the model has its own independent set of properties. For dimensions and auxiliary coordinate constructs, the allowed properties are `axis`, `calendar`, `leap_month`, `leap_year`, `long_name`, `month_lengths`, `positive`, `standard_name`, `units`. Auxiliary coordinate constructs of time are optionally climatological; this property is indicated by the presence of the `climatology` attribute. In any field, any given value of the `axis` attribute can occur no more than once among all the dimension and auxiliary coordinates of that field. The CF data model allows field, dimension and auxiliary coordinate constructs to have other properties not defined by CF, provided they do not conflict with CF, but since they are not part of the CF standard, the data model does not provide any interpretation of them.

Karl Taylor 6/23/12 9:31 AM
Comment [11]: Can't the global attribute simply be augmented, rather than superseded by the variable attribute (e.g., the "comment" attribute)?

Karl Taylor 6/23/12 9:22 AM
Deleted: it is superseded if

Karl Taylor 6/23/12 9:21 AM
Deleted: its own

Karl Taylor 6/23/12 9:25 AM
Deleted: C

The attributes `valid_max`, `valid_min` and `valid_range` of data variables and coordinate variables are checks on the validity of the values, which could be verified on input and written on output. In this CF data model we assume they do not constrain any manipulations which might be done on the data in memory, and they are not part of the data model.

The attributes `_FillValue` and `missing_value` of data variables specify how missing data is indicated in the data array. This data model supports the idea of missing data, but does not depend on any particular method of indicating it, so these attributes are not part of the model.

Karl Taylor 6/23/12 9:33 AM
Comment [12]: I think this is a mistake. The "missing_value", at least, should be part of the data model and included when needed in the field construct. Otherwise how can software know which values of a field it should operate on and which are bogus and can be ignored?

The attributes `add_offset`, `compress`, `flag_masks`, `flag_meanings`, `flag_values` and `scale_factor` are all used in methods of compressing the data to save space in CF-netCDF files, with or without loss of information. They are not part of this data model because these operations do not logically alter the data, except that the `compress` attribute implies two alternative interpretations of coordinates (compressed or uncompressed). The "feature type" attribute and associated new conventions, to be introduced in CF version 1.6, will provide a way of packing multiple fields of the same kind of discrete sampling geometry (timeseries, trajectories, etc.) into a single CF-netCDF data variable, in order to save space, since a multidimensional representation with common coordinate variables is typically very wasteful in such cases. This is a kind of compression. The data model would regard each instance of the feature type as an independent field construct. However, the "feature type" attribute itself is also a metadata property that would be a property of the field construct and part of the data model.

The attributes `bounds`, `cell_measures`, `cell_methods`, `climatology`, `Conventions`, `coordinates`, `formula_terms` and `grid_mapping` have various special or structural functions in the CF-netCDF file format. Their functions and the relationships they indicate are reflected in the structure of this data model, although?? these attributes do not correspond directly to properties in the data model.

Karl Taylor 6/23/12 9:30 AM
Deleted: and

6th February 2012
Version 0.2 of 1st August 2011
Original version 0.1 of 10th January 2011

Jonathan Gregory and David Hassell