Geosci. Model Dev. Discuss.,
https://doi.org/10.5194/gmd-2017-154-RC1, 2017

Geoscientific Model Development Discussions

Interactive comment on "A CF data model and implementation" *by* David Hassell et al.

V. Balaji (Referee)

balaji@princeton.edu

Received and published: 4 August 2017

The paper is an in-depth discussion of the CF conventions, which are becoming indispensable for producing climate data conforming to a set of agreed-upon standards. They ensure that software can be built that in principle can process any conforming dataset.

The authors go further and formalize the underlying data model. The data model is an exact and comprehensive formal representation of every possible facet of the CF conventions.

Further, and finally, the paper describes cf-python, a reference implementation of the data model. Software built upon cf-python is guaranteed to be able to express and process any possible CF-conformant dataset.

As far as I can tell, the description of CF is precise and complete; the data model is an utterly faithful replica; and I have no reason to doubt that the software is as well.

I have little hesitation in recommending for publication. The bulk of the content of the paper (Sections 1.1-5.3) requires very little revision. Beautifully written and rigorously edited... no typos or grammar errors I was able to find.

I would however require some revisions to the beginning and end of the text (Sections 1 and 6). I feel the need for a data model isn't sufficiently well motivated; the current problems and anxieties surrounding CF aren't well laid out. Nor is it clear from reading the article how CF may evolve in the future, and what the CF data model will do to guide this evolution. With these changes, I think this paper stands a good chance of becoming a canonical citation for CF, much the same as the Nativi et al paper is for the Common Data Model underlying netCDF..

Let me pose a few questions, and let's see if the answers could help guide the kinds of modifications I would suggest.

1. The data model has been written to be independent of the underlying encoding (L74), viz. netCDF. Nonetheless, the data model is written to conform as far as possible to the netCDF classic format (L83). There have been quite some discussions on whether to introduce new features in CF that in fact require a departure from the classic format. People have pointed to the lack of strings in netCDF-classic (see L92 and L198 in this paper), and whether the use of "groups" and other features imported from HDF into netCDF4 would in fact be a good thing. There are strong opinions on either side of this (including mine..) but without adjudicating this issue, do you think the abstract data model is versatile enough to be forward-compatible (as well as backward-compatible, L29)?

2. There are already features emerging that break the dependence on files, treating "atomic datasets" as synonymous with files. This is likely to be even more so in the feature, as technologies move away from filesystems to object-store. Could you comment

on how the data model will enhance, or inhibit, such a migration? Examples include the "external_variables" attribute that allows the referent of a cell_measures attribute to be in a different, unspecified, location. How would the data model, and cf-python, deal with such cross-references?

3. There have been proposed extensions to CF including Gridspec and UGRID for unstructured grids. Is the data model up to the task of representing such features as say, grid mosaics, which may involve more cross-file references? Or are such concerns entirely outsourced to ESMF and not of concern to the CF data model itself?

4. In general, it has become hard to evolve CF. This is not a problem with CF itself, rather with the sociology of standards: changes require a sufficient number of people to pay sufficient attention in a reasonable amount of time. If the CF data model became the canonical representation, and cf-python the reference implementation, one approach would be to make all proposed changes visible in the data model, and perhaps we could also require the proposer to submit a pull request implementing the suggested change in cf-python. This would give the debates greater clarity; but would they make the problem of evolving the conventions even harder?

5. If the conventions don't evolve rapidly enough, usually small communities break away and work on their own "fork" of the conventions (see UGRID). This could become explicit using the data model: in fact this would mean an actual fork of cf-python, which could make it back on the git trunk at a later date, or not at all. Would this be a good outcome?

I believe some discussion of these questions would greatly enhance the paper.

Minor comments below:

L125: missing value and fill value aren't the same thing. It might be convenient to set them to the same value, but I can think of use cases where you would not want that.

L810: more than reducing precision, this reduces to the representation to integer. Thus

dynamic range is limited (packing to 8-bits means the lowest and highest values after the offset have to be within a factor of 256), and values are discretely spaced.

L819: Currently the netCDF4 library has lossless compression based on zlib... others (Dennis et al, https://www.geosci-model-dev.net/9/4381/2016/gmd-9-4381-2016.html) are proposing lossy compression.

L19, L34, L128, L318, L660, L675, L745, L776, L778, L859, L861, L869, L875, L876: convert to hyperlink (use hyperref package) L36, L140: use \equiv instead of = for definitions L126, L169, L172, L174, L179, L189, L195, L274, L299, L306, L327, L328, L334, L335, L337, L341, L342, L370, L371, L375, L377, Fig 11 caption, L570, L811, L841: for CDL entities in the main text use \texttt{} instead of quotes. L673,L872: omit "Ryan".

Interactive comment on Geosci. Model Dev. Discuss., https://doi.org/10.5194/gmd-2017-154, 2017.