Thoughts on Proposals of Structural Changes to GRIB

TOYODA Eizi (Japan)
toyoda(at)npd.kishou.go.jp

17 May 2012

Basically I welcome ideas of two documents submitted by Dr. Enrico Fucile of ECMWF to reform management and structure of GRIB into more computer- and user-friendly manner. This would be epoch-making things. I just wanted to add some pieces to make it really beneficial for potential stakeholders.

1. GRIB Template Tables [Doc 2.1(2)]

(1) Concept "type" is overloaded

Frankly, I wonder "what is definition of 'type' in this table?" Typical definition in computer science is "mapping between octet sequence and numerical or character string value."

- That fits perfectly for types: signed, unsigned, ieee_float, char, and code_table
- But not for types: repetition and repetition_count

A problem for programmer's viewpoint is that "repetition" is parameterized: it must come with name of variable like "repetition(n)". That means the column "type" is actually free form text including parenthesized mathematical expression, which programmer must parse....

(2) Concept "table" is overloaded

Similarly, I tend to compare the concept of "table" here to common relational table or Excel table. I find:

- Order of rows has meaning here, unlike relational table
- Some of rows have special role to modify meaning of others
- Octet position matching to a row has to be calculated by complex equation that require many times of sequential scanning of the table

We have claimed GRIB as "table-driven". Why don't we adopt common definition of table? By doing so, proven strength of tabular structure will empower us in every aspects of programming and data management.

(3) Context-dependent interpretation of numbers

Minor thing, but confusing or irritating for programmers:

- Column “Octet numbers” may be either number or range. If the table is literally implemented so, programmers have to prepare for both cases. But the end of the range is not necessary, since it can be calculated from “size”.
- Column “size” is number of octets for most rows. But it refers to number of rows for “repetition” type.

(4) Suggested table structure

Basically what I suggest is commonly called database normalization.

- A column must contain “scalar value” (such as number or string) which is not analyzed anymore
- Complex structure such as list, range, or mathematical expression should be decomposed into combination of columns in number type

The question is how to handle expression to compute octet position, but we can assume limited complexity for existing GRIB templates.

Every existing GRIB templates in Sections 3, 4, and 5 have no more than two repetitions with one or two fixed content sandwiched in them.

$$\text{Template} := \boxed{\text{fixed}} + n_A \times \boxed{\text{repeat A}} + \left(\boxed{\text{fixed}}\right) + n_B \times \boxed{\text{repeat B}}$$

For each element, we can calculate octet position by:

$$\text{Pos} := b + w_A \times (i_A - 1) + f_A \times (n_A - 1) + w_B \times (i_B - 1)$$

Where b is named after base position, n_A and n_B are count of repeated part A and B stored at positions p_A and p_B respectively, w_A and w_B are width of repeated part A and B respectively, f_A is fixed shift width after repeated part A, i_A is variable index (satisfying $1 \le i_A \le n_A$) for element within repeat part A or 1 for others, and i_B is defined similarly.

The table becomes like following:

Tem-plate	*b*	p_A	w_A	f_A	p_B	w_B	type	size	content
4.14	10	—	0	0	—	0	Unsigned	1	*[Fixed part]* Parameter category
4.14	11	—	0	0	—	0	Unsigned	1	Parameter number
…	…						…	…	…
4.14	54	—	0	0	—	0	Unsigned	1	NC – number of forecasts in the cluster
…	…						…	…	…
4.14	72	—	0	0	—	0	Unsigned	1	n – number of time range specifications describing the time intervals used to calculate the statistically processed field
4.14	73	—	0	0	—	0	Unsigned	4	Total number of data values missing in statistical process
4.14	77	72	12	0	—	0	Unsigned	1	*[Repetition part A]* Statistical process used to calculate the processed field … (see Code table 4.10)
…	…						…	…	…
4.14	85	72	12	0	—	0	Unsigned	4	Time increment between successive fields, in units defined by the previous octet
4.14	89	72	0	12	54	1	Unsigned	1	*[Repetition Part B]* ensemble forecast number

Six columns for location may look ugly for traditional eyes. But it has benefits from programmer's and code maintainer's viewpoints:

- Every rows have equivalent roles; we don't have to scan entire table to check special rows
- Every column has simple type: string, integer, or enumerated type name. They are simple enough that all programming language or database has natural expression. [Note: dash (—) in integer columns p_A and p_B are intended to be NULL, but can be changed to any dummy number because zero is multiplied later.]
- No two rows have the same number in column *b*; we can simply throw all templates into single relational table, and then select desired template name and sort by *b*.
- Order of rows doesn't matter at all; if the table is amended between versions, we can just record the change, not where the change is
- Octet position for a row is calculated by selecting three rows in the worst case (one for the row itself, and then twice to retrieve sizes of n_A and n_B); no scanning is needed to determine whether the row is within the scope of other repetition row

(5) Footnotes

How do we handle footnotes? I suggest that a new column with very large limit on length in a computer-readable representation of template table. We must not be afraid of having several footnotes with the same content. Benefit of self-description outweighs compactness.

2. GRIB Edition 3 [Doc 2.2(3)]

(1) Originator

In addition to traditional organization-based identification number pair (centre and sub centre), a description of project is added as mandatory. And further extension is possible using templates. Am I taking right? It is fantastic.

I wonder what kind of project is eligible here. More frankly, how many projects will we have by the end of lifetime of this new GRIB structure?

We can work with single octet identifier for project, if we impose very strict burden such as "only Congress-recognized project is eligible". But I suspect that will frustrate most of potential users.

Contrarily, if we loosen policy like "anybody can request project number", single octet will be quickly filled up. This is more than question of the number of octets. Does the WMO community really want rigorous identification of thousands of projects, at the cost of busy registration procedure? Are the academic people really happy with waiting for half year for fast-track? The benefit of centralized management of code table is proof for strict "no-collision" requirement, but I suspect it doesn't apply for larger community. For such people adequate length (say 64 chars) of free text field might be more preferable.

Another viewpoint: interoperability between GRIB and CF-NetCDF is a long-standing issue. From viewpoint of operational centres that had invested on tooling for GRIB, the question is how they can quickly import new data in CF-netCDF or HDF. Organization is identified at free text ":institution" attribute. For ad-hoc purposes (like R&D) it is enough to simply copy that string, without waiting for months for fast-track registration.

So my suggestion is to develop an Originator Definition Template having a free text field to specify data source (it may also include numbers). I don't oppose code table for project , and free text feature doesn't have to be default. But it would be unfortunate for software development that I will have to do that anyway in local table.

(2) Multi-dimensional structure

It is very exciting to have multi-dimensional array in a dataset. I strongly support the idea of extending "dimensions" from spatiotemporal 4D (time, latitude, and longitude, height) to include process and parameter.

Inside Japan's NWP system, grid data is managed by a kind of database called NuSDaS. I've written a brief introduction to 2005 AMS meeting <URL:http://ams.confex.com/ams/Annual2005/techprogram/paper_87457.htm>. The key thing is it's a database of two-dimensional array data identified by keys in six "dimensions":

- Dataset name (16 letters string),
- Reference date-time,

- Member name (of ensemble forecast),
- Valid date-time,
- Level name, and
- Parameter name (6 letters string).

So a dataset for a reference time contains 4-D matrix (multiple member × valid-time × level × parameters) of 2-D horizontal grids. It is very good if we could convert it into single GRIB3 message.

A piece of knowledge from experience: the 4-D matrix is often ***very sparse***. That may be due to physical limitation or wish to save bandwidth. Examples:

- Level-parameter dependency
  - pressure reduced to MSL only appears on MSL
  - important parameter at all levels, others only at 500 hPa
- Valid-time-parameter dependency
  - precipitation does not appear in initial
  - important parameter 3-hourly, others only 24-hourly
- Member-parameter dependency
  - people want to package raw fields for each members and statistics

NuSDaS uses sequential data records with 4D rectangular array index having octet offset inside a file. I don't say GRIB has to do the same implementation, but it's worth considering how we deal sparseness.

(3) "Principle of separation of data and metadata"

I'm afraid it's the first time to see this phrase. I'm not objecting about the specific 10-section structure proposed for GRIB3 without repetition, but I feel awkward to establish this "principle" without knowing well. It's just a personal feeling but I'm afraid of this might grow into another monster.

Please note: I agree and support enthusiastically that we have functional demand that listing of structure must be effective. This is also my experience. This structure looks good so far. I just want to have clear reasons for choices.

I have to mention on limitation of the proposed metadata-data structure. Once we complete metadata part and start writing data, it is no longer possible to expand axes. List of coordinate values (times, parameters, or whatever) is written inside the metadata part, so we have to rewrite entire message to expand any dimension.

That is not a problem at all for operational NWP. But R&D people often come to me to request that they don't want to prescribe forecast time hours or list of parameters at the beginning of a model run. Indeed netCDF allows one dimension may be extendable, which is usually assigned for forecast time. In my opinion GRIB exists to serve operational meteorology but I'd like to share that for consideration.

(4) Time interval issues

I agree we must have clear regulatory document, especially that describe units of measurement for every case of time templates. WMO Code exists to deliver information between people. Since number and units of measurement are essential parts of physical quantity, a code failing in clear delivery units is totally failing in its job.

I think it's a good idea to have two basic templates for instantaneous and interval values. We should have simple structure that can describe often-used information without complex manipulation of abstract concepts.

I guess your time template for interval value should not change units described in the table of parameters. It's good thing. But we have to be aware that we must allow time-integrated parameter (such as precipitation [kg.m-2]) in addition to tendency (such as precipitation intensity [kg.m-2.s-1]). So we must relax the orthogonality regulation 92.6.2 to some extent, at least to allow such pair.

I don't mean we can just abolish 92.6.2. It has its role, and works well in some other aspects. For example, now that "maximum temperature" is deprecated, people are directed to use statistics template "maximum", and that ensures we have metadata i.e. time interval for maximum. The regulation was introduced to ensure that each GRIB message comes with adequate metadata through enforcing time-related product template.

So the question is whether we continue that attitude on time-interval quantities or not. Is it okay that people use instantaneous template with parameter "precipitation" and time range gets unclear? We can take two ways.

Paternalistic approach: the parameter table has new column named "interval flag". If that is true (or 1, up or whatever), the parameter is not allowed to use with instantaneous template. We can construct validation software to issue warning for undesirable usage.

Laissez-faire approach: if above is considered ugly, we can simply declare that it is responsibility of GRIB-originating centre that GRIB messages comes with appropriate metadata wanted by recipients. Actually CF-NetCDF is in this way. That means (1) at least R&D community is fine with this and (2) we have to care about importing such data.

(5) Horizontal "grid"

I think it is a good opportunity to re-define the word "grid". In geospatial technology it is defined as something like "grid or raster is a set of points defined as Cartesian product of equally-spaced lists of coordinates in some coordinate system". This pure definition still works for most of nowadays GRIB practices. But more and more number of exceptional grid systems appear: thinned grid, spherical harmonic coefficients, reduced gauss grid, triangular icosahedral grid, and hexagons today. So it should be difficult to say more than "grid in GRIB is a set of points defined by some simple expressions".

Please note I don't mean we should limit such inventions. Opposite. We must show some clear definition of "grid" at the top of regulations to avoid being misunderstood by geospatial professionals.

(6) Vertical dimension

Damn question. Can we safely assume there is a vertical dimension? That is true for most grid data for atmosphere and oceans. But we can't define vertical axis for Geocentric Cartesian coordinates of geodesy or Heliocentric Cartesian coordinates of space weather domains. I know it is really edge case for GRIB community, so it is fine for me to define conventional third axis to be described in Vertical Coordinate Definition Section.

(7) Meanings of "property"

I'm sorry for being trivial. The proposal uses the word "property" as synonym to "metadata". No problem. But it is a bit confusing to see "the ***properties*** defining time, space, process, ***property***, and data" (at the beginning of section 4). I'd be very happy if we could consider and define (and perhaps invent) clear terminology.

(8) Parameters

I'd like to have well-documented criteria to add a new parameter. That will help us avoid lengthy discussion. I'm thinking of something like following. (Not exhaustive, details can be changed)

- Parameter name should not be jargon in one discipline
  - Even in an unfortunate case that universal word cannot be found, the name must make sense at least to proposing discipline and data management community.
  - IPET-DRC may request rewording if a proposed name doesn't help data management people to identify and differentiate parameters.
- Parameter must be unique
  - Proposal is not accepted if there exists another parameter with the same meaning , regardless the discipline
  - If there is a need for specialized or generalized word, that is okay (ex. "precipitation" doesn't prohibit "convective precipitation" and vice versa)
- Orthogonality to temporal statistics
  - A parameter must not represent a physical quantity as a result of temporal statistics of another parameter (base parameter). Users are advised to use temporal template representing such statistics (ex. "maximum temperature" should be "temperature" with time template representing "maximum over time"). If base parameter is not yet present in the parameter table, new one is created.
  - Nevertheless it is no problem to register a physical quantity defined as a time integration of another parameter.
- Orthogonality to vertical limitation

    - A parameter must not represent a physical quantity with vertical limitation. Similarly to time statistics, “Temperature at 500 hPa level” has to be decomposed into “temperature” and vertical template specifying 500 hPa level.
    - If such decomposition is difficult, it is okay to have words meaning vertical position. Ex: “low cloud”, “middle cloud”, and “high cloud” are tolerated because they are recognised words in synoptic reporting based on morphological classification, and hence have no height to describe.
- (list continues)

It should also be worth considering relationship with “standard name” table of the CF convention. Some possibilities:

- Partial linkage: GRIB3 parameter table has a new column referencing to standard name if applicable
- Total unification: standard name becomes mandatory. Proposers must get standard name first and then apply to IPET-DRC to obtain numbers. As a result IPET-DRC won’t be troubled about identification of parameters.
- No ties: nothing to think, but software developers have to make isolated parallel efforts to identify mappings

There should be some differences between GRIB2 parameters and CF standard names. Direction of vector components is an example I know.

- GRIB Editions up to 2 uses bit flags to describe directions of vector components.
- CF registers separated standard names for different directions: ex. “eastward_wind” and “x_wind”.

Unfortunately I have to say CF approach is far easier to understand in this case, but there might perhaps be a case that GRIB community is not happy to follow CF handling.

(9) Data representation

Sorry again for being trivial:

- The word “processing” also appears in Section 5 Process definition section.
- I’d be very happy if we could have good definition of “data representation”. My definition is “mapping from octet stream to values on a grid”. It is important to know which is “from” and which is “to” (or whether we are thinking of encoding or decoding), since the words “pre-processing” and “post-processing” have reversed meanings.
- I don’t understand why bitmaps is independent section. In my understanding it has to be a kind of post-processing (or pre-processing, depending on definition). As an independent section it becomes mandatory. It’s a waste of space, isn’t it?

(END)