EDOC114

EUROGAM PROJECT

Comments about EDOC061 Unified Spectrum Format and EDOC102 Spectrum Database Structure (csnsm-ipn header format)

Edition 1.0

january 1991

H.HARROCH, B.MERLANT, M.DZIRI
IPN-CSNSM (Orsay)

Unified Spectrum Format describes a proposed format header for spectra. A spectrum is made up of a set of histograms or matrices, calibrations and annotations, as defined in EDOC060(D.BRIGHTLY).

Comment 1

There is a set of histograms or matrices, up to height. One main histogram (or matrix) and the others derived from the primary one. All must have the same "shape". I aggre with the idea that there are derived histograms or matrices. Noise histogram is given as an example. But, I think that derived histograms (or matrices) may have not the same "shape". For example, with a two dimensionnal matrix, a slice or a projection have only one dimension. Accordingly, the Data Array Descriptor must have a dimension field.

Comment 2

There is no field to indicate the number of data arrays.

Comment 3

The proposed format is very close to a run-time implementation in memory. So, it is not very suitable for stored spectra. I agree with the idea of an unified spectrum format. But unified spectrum format, means, that both on-line and off-line software handle the same spectrum header on disk. a large variety of programs may handle the header spectrum, so no assumption has to be made on the header specification about implementation .

Comment 4

The spectrum format must make easy data exchange. So all infor- mation in the header must be in ASCII.

Comment 5

The header format must be extended easily. So extension mecanism has to be provided.

The header has to be made up of two part:

- a fixed length part, containing all necessary informations.
- a variable length part containing optional informations.

fixed and variable parts of the header are in ASCII. An item of the variable part begins with a field containing the extension-id field, telling the purpose of the item which is added to the header spectrum. Following field depends on the extention-id. (see a proposed format below) This sheme in derived from technics used in packet switching software (X25 network) where optional services are implemented in X25 header following this scheme.

Comment 6

IN2P3 header has a very simple format. it can only have one data array. All informations contained in the IN2P3 format are in the proposed Unified Spectrum Format except byte ordering and total header length. The IN2P3 header could be extended by increment of 256 bytes. So new fields like dimension or annotation could be implemented. But, it would be better to defined a true extensible header. So future fonctionnalities would be easier to implement.

a fully extending header spectrum format

The header is made up of three parts:

1. a fixed length part
2. a variable length part
3. a optional part

- **part 1:**
  part 1 contains general informations and all necessary informations about part 2 and data arrays.

. application-id	”SPECTRUM-HEADER ”	(16 char)
. header version	”VERSION-xxx ”	(12 char)
. spectrum name	”Ge1 ”	(80 char)
. creation time	”nn-mmm-yy hh:mm:ss ”	(20 char)
. modification time	”nn-mmm-yy hh:mm:ss ”	(20 char)
. experiment name	”GAMMA-GAMMA ”	(80 char)
. run name	” ”	(80 char)
. run number	”xxx ”	(8 char)
. number of data array	”xxx ”	(8 char)
. byte ordering	”MSB ”	(4 char)
. total header length	”xxx ”	(8 char)

  free space up to 512 characters, for future extension or a general comment.

- **part 2:**
  part 2 is made up of data arrays descriptors.

data array descriptor format:

"ArrayID ArrayNumber ArrayName ArrayType DataType MinValue MaxValue DimmensionNumber base1 range1 base2 range2 ...."

field 1: item identifier ArrayID
field 2: data array number
field 3: data array name. Each data array may have a different name.
field 4: array type.
    "matrix "
    "half-matrix "
    "graph16 "
    "graph32 "
    "graph "
an histogram is a matrix with the dimension field set to 1.
field 5: data type.
    "INT*2 "
    "INT*4 "
    "REAL*4 "
field 6: minimun value
field 7: maximun value
field 8: dimension of the data array
    "1D " "2D " ....
field 9: base 1
field 10: range 1
field 11: base 2
field 12: range 2
field ...

- **part 3:**

part 3 is made up of optional items. An Item is a character string terminated by a null character. Each field in an item is ended by one or more blank chararacters. The first field of each item is the identifier of the item (ItemID).

example:

"CalibrationID 3 1 value1 value2 ...."

field 1: item identifier CalibrationID
field 2: data array number 3
field 3: dimension array number 1
next fields: information fields

following this scheme, any new item can be added.

- **annotation:**
  "AnnotationId HistogramNumber DimensionNumber InfoToDimmension1 "
- **calibration:**
  "CalibrationId HistogramNumber DimensionNumber ....."
- **efficiency:**
  "EfficiencyId HistogramNumber DimensionNumber ....."
- **information:**
  "InformationId HistogramNumber DimensionNumber ....."
- Other item ....