Ideas for Information Types and Metadata Attributes — Jim Almond, CHPC — Second Try: 2/94

This is an update of the paper I distributed before Christmas, expanded to include further possible Metadata items.

There seems to be basic agreement that metadata can be organized into I) those describing the informational entity being stored, and II) those pertaining to its storage and use. On this basis, I propose the following organization and terminology:

I. **Metadata describing the informational entity itself**
These descriptive attributes allow users to identify and browse entities without accessing the entity itself, and to access information more intelligently and selectively. They also support the capability for applications to exercise typing constraints, automatic type and format conversions, and to optimize computations on stored entities *at run time*.

1. Metadata attributes pertaining to the *Abstract Data Type* (ADT) of the entity. This generalization of the concept of "data type" includes such descriptors as:
    a. The designation and description of the ADT. This specifies the *Informational Identity* (IId) of the entity, and thus its functional semantics — but not its physical representation or "format".[1]
    b. The ADT(s) of the *elements* of the entity, such as real numbers, integers, coordinates, vectors, or complex ADT's.
    c. Attributes specifying the *logical* structure of the entity. For regular structures consisting of multiple elements of common ADT (or type), such as the topologically Cartesian structures of matrices and grids, the logical structure can be described implicitly in terms of attributes such as a *dimensionality* and the number of elements, band width, etc. in each dimension. For other structures, such as unstructured grids or databases, the irregular relationships among elements which comprise the logical structure must be described explicitly, often in an application-specific manner. Fortunately, standards are being developed for interchangeable logical structures in some areas.[2]
    d. The mapping of the elements of the logical structure to a physical structure in the one-dimensional address space of computer storage media. For regular structures such as topologically Cartesian grids and matrices, this can be specified in a standard manner, e.g. by indicating storage by rows or columns, and whether logically zero elements outside the band width are physically stored or not. For more general structures, such as databases or unstructured grids, the mapping will usually take a form defined by the specific software system creating the entity. Thus, the identity of the creating application may be an important attribute.
    e. The encoding of individual elements in the binary language of the computer, possibly including compression and encryption techniques. In this sense, the identity of the hardware on which the entity was created may be an important attribute.

[1] The list of ossible ADT's is essentially endless, and arbitrarily complex, such as: molecular structure, electron density map, student record, bank statement, lunch menu, bus schedule, road map, insurance claim, budget, balance sheet, recipe, player roster, mailing list, meeting agenda, itinerary, scientific article, book, weather forecast, Computer Graphics Metafile, IGES product description — — — .

[2] Some examples include finite element models, and the ANSI X12 effort on Electronic Data Interchange (EDI) in administrative areas.

2. Metadata attributes describing *Informational Essence* contained in the entity. Some examples are:

    a. Description of the meaning of the entity or its elements, usually entered *explicitly* by the creating process which "knows" such information, or by direct human action. e.g. "the elements in this grid are temperatures in units of C°", or "this image shows an X-ray image of Jon Doe's left lung on 2/4/94", or "this structure represents human hemoglobin".
    b. Information *derived* by processing the entity, such as pattern recognition, keyword search, or by direct human action, e.g. "this X-ray image shows 3 incipient cysts".
    c. A compressed image or "Icon", representing the entity, and facilitating human recognition of it.

3. Metadata attributes describing relationships between the entity and other entities. Some examples include:

    a. *versions*, in the sense that one entity (the son) was derived from the other (the father) by modification,
    b. *set* relationships, indicating that informationally congruent members belong to groups such as time series, spatial subdivisions, etc., which may for instance be advantageously stored physically near each other in the interest of efficient collective access,
    c. *hierarchical* relationships, such as entities which are elements of higher entities, e.g. in support of systems offering varying levels of detail,

II. **Metadata pertaining to the storage and use of the informational entity**
These attributes support the control, administration, and optimization of the actual physical storage of informational entities.

1. Metadata attributes pertaining to the storage and replication of entities on physical media. (This model does not assume entities are mapped 1:1 to files, although it will probably be so in most cases.)

    a. Pathname(s) pointing to file(s) which comprise the entity
    b. Date and time of last access, creation, last metadata change
    c. Size
    d. Accessibility requirements or other means of influencing storage and migration strategies
    e. Physical storage location of the most easily accessible instance(s) of the entity, and of additional instances
    f. Replication (backup) requirements for reliability

2. Metadata attributes recording or controlling the use of entities.

    a. Access control lists by UID, by account, by hardware, by network location, time, etc.
    b. Default access mechanism (stage copy to local disk or direct access of remote elements)
    c. Log of individual accesses and times by UID, by account, by hardware, by network location, etc. (for statistics, security, and/or accounting).

Examples of ADT's and their Attributes

Since the set of attributes appropriate for describing an informational entity as proposed in category I is dependent on its ADT, there will be a wide variety of such metadata, depending on the requirements of the discipline and the whims of application developers. Furthermore, such "metadata" have been traditionally attached to and stored with the entity in the form of "headers". This common practice has two clear disadvantages:

- When and entity is migrated or archived to a less accessible medium, the descriptive information in the header also loses accessibility.
- Differing header formats for otherwise identically stored and encoded entities are a common cause of incompatible "data formats".

It is hoped that clearly defined metadata concepts can help overcome these obstacles.

As an exercise to clarify and test concepts, I have constructed a number of tables showing examples of ADT's, with possible attributes organized according to the scheme presented above. Although not meant to be complete or definitive, they can perhaps suggest a coherent scheme for the definition of informational entities and their metadata attributes. Storage-oriented metadata in category II are omitted for simplicity.

Metadata Attributes for some sample informational entities							
1a: Specification of the ADT	**Semantics**	**1b: *logical* structure**	**1c: ADT of elements**	**1d: Storage Mapping (*physical* structure)**	**1e: Element encoding**	**2: Informational *essence***	**3: Relationships**
a) Vector		Length L	complex numbers	Same as logical sequence	IEEE standard	Meaning, units, label	
b) Square Matrix	Defined by the rules of matrix algebra (can be inverted, transposed, or multipied with other conforming matrices)	N = 1200; band B=250	reals	by column; full matrix stored	IEEE F.P. scalars	largest element = 12.6	these two are in the same *family*, and should be stored near each other
		N=B=240; Symmetric		by row	ASCII values	CN = 10**6	
		N = 1200; upper triangular, band B=250		by diagonal; only upper band stored	Cray F.P.		these two were derived from the LU decomp. of the first entity
		N = 1200; lower triangular, B=250		by column; only lower band stored			

1a: Specification of the ADT	**Semantics**	**1b: *logical* structure**	**1c: ADT of elements**	**1d: Storage Mapping (*physical* structure)**	**1e: Element encoding**	**2: Informational *essence***	**3: Relationships**
c) Set of data columns		N columns	columnms with M elts	ordered by column	for each column	meaning & label of each column	
d) A 2-D CAD/CAM drawing		Defined by the CGM standard			cleartext		
e) 3D cartesian grid	-Map to 2D -Direct vol vis'n	M=N=256, L=1000	reals	in M,N, L order	IEEE F.P.	temperatures in C°, Max=6C°	

1a: Specification of the ADT	**Semantics**	**1b: *logical* structure**	**1c: ADT of elements**	**1d: Storage Mapping (*physical* structure)**	**1e: Element encoding**	**2: Informational *essence***	**3: Relationships**
f) 3D Cartesian spatial grid of x,y,z coordinate triplets	-Map to 2D -Spatial vis'n	256x512x512 (x,y,z)	integers	all x's, then all y's, then all z values;	Cray integer format	Max and Min values in x,y,z	
g) 3D Cartesian grid of scalars	-Map to 2D -Direct vol vis'n -Map to conforming spatial grid	256x512x512 (x,y,z)	reals	all z's, then all y's, then all x values	Cray F.P. values	Stresses at P=21.5, in	these map to the previous coordinate grid f)
h) 3D Cartesian grid of scalars						Strains at P=36.8	
i) 2D Cartesian grid of scalars	-Visualization	256x512	(x:y:real) triplets	x,y order	IEEE Int's and F.P.	Strains @ P=36.8, z=4	2D slice of (h) mapped to f)

1a: Specification of the ADT	**Semantics**	**1b: *logical* structure**	**1c: ADT of elements**	**1d: Storage Mapping (*physical* structure)**	**1e: element encoding**	**2: Informational *essence***	**3: Relationships**
j) (2D) digitized raster image	Vis'n	2048x2048	pixels in RGB space	Stored by scan line (row)	each pixel has 24 bits for RGB values; jpeg compressed	Digitized X-Ray; shows 3 cysts; icon of image	this is a "before-after" pair
k) (2D) digitized raster image						Digitized X-Ray; cysts reduced; icon of image	
l) technical report with logical content.	Logical operations,, e.g. on "chapters", spellchecks, etc.	Created by an SGML editor; elements are chapters, paragraphs, etc. as defined by specific DTD grammar			ASCII text strings	Title, keywords, DTD designator, recommended FOSI; icon image of cover page	"Father" version of m)
m) same as l), with modifications							"Son" of l)
n) same as m), formatted to postscript	can only be printed	as defined by Postscript				Title and keywords, DTD, FOSI	Derived by formatting m)

1a: Specification of the ADT	**Semantics**	**1b: *logical* structure**	**1c: ADT of elements**	**1d: Storage Mapping (*physical* structure)**	**1e: element encoding**	**2: Informational *essence***	**3: Relationships**
o) A C++ source program	can be compiled, edited, printed, etc.	ASCII text strings					
p) A C source program		ASCII text strings				Named MYPROG	derived from n) using CFRONT precompiler
q) Object code for SUN Sparc hardware	can be loaded and executed on Sparc	Binary in SUN Sparc format				compiler options used	derived from p) using C compiler
r) Student record database		X12 EDI Standard		As defined for SYBASE	Binary SYBASE object	50,001 students for 1992	
s) Student Record from database						Name, courses, grades, etc.	element of r)

A Possible Taxonomy and Attribute Organization: Example for ADT *Square_Matrix*

The systematic organization of a set of ADT's and their attributes is a forbidding task. As a learning and demonstration effort, I have tried to approach the task using a well known and very regular structure, whose semantics are well defined — the square matrix.

1. Metadata describing the ADT

   A. ADT = *Sq_Matrix*. This essentially specifies the semantics of the entity according to the rules of matrix algebra, independent of the remaining attributes.

   B. ADT's of the matrix elements. Typically, these will be standard data types such as integers, real values, complex numbers, etc. Note that this does not specify the encoding of the values; a real number can be encoded in ASCII text or in some floating point binary format, etc.

   C. Logical Structure: These attributes may refine or limit the semantic operations, and offer a basis for possible optimizations.

      a. The *Form*[3]

         i. General matrix
         ii. Hermitian
         iii. Lower triangular
         iv. Upper triangular

         Note that the significance of logical structure attributes is in no way affected by the physical structure attributes. For example, the definition of forms with systematically zero-valued elements, such as lower or upper triangular, does not specify whether these elements are actually stored or not.

      b. Dimensions

         i. The dimension N of the matrix
         ii. The logical half-band width B (Outside this band, elements are systematically known to have a zero value. However, they may still be stored or not, as specified by the *physical* structure below.

   D. Physical Structure

      a. Storage order mapping

         i. by row
         ii. by column
         iii. by diagonal
         iv. by explicit index specification (for sparse matrices)

---

[3]

b. Physical storage of zero elements or derivable elements

i. Indicator (Full_form) for the storage of derivable elements of Hermitian or Lower/Upper triangular matrices. For example, this indicates whether the zero valued elements above the diagonal of a lower triangular matrix are explicitly stored (True), or not (False).

ii. Indicator (Full_band) for the storage of elements outside the logical half-band width, indicating whether they are explicitly stored (True), or not (False — in which case any element i,j with j>i+B-1 is assumed zero, and is not stored).

E. The encoding of the elements in the binary format of the computer.

2. Informational essence of a matrix can include such things as its eigenvalues, rank, trace, condition number, physical meaning, units, etc,

The following table depicts distinct combinations of attributes of square matrices with respect to:

- logical *Form*,
- Full_form, (indication as to whether the matrix is fully stored regardless of form,
- physical storage order mapping,
- half-bandwidth B

Simplifications in the table include:

- Assumes elements outside the band B are *not* stored (Full_band is always False).
- Includes only systematic Cartesian orderings; (does not include storage using explicit indices).
- Does not consider the various encoding techniques for the elements.

The table shows 42 combinations of these attributes; even a unary operation on such a matrix could require 42 variants! Thus even without considering encoding variants, a binary operation such as the multiplication of two conformable square matrices, could require 42**2=1764 variants!! Obviously, strategies are needed for simplifying or systematizing such variation. Any ideas?

Form	Full_form	Storage by row		Storage by column		Storage by diagonal	
		B = N	B < N	B = N	B < N	B = N	B < N
General	T						
Hermitian	T						
	F						
Lower Triangular	T						
	F						
Upper Triangular	T						
	F						