the brain by use of NMR and PET data.

ACKNOWLEDGMENT

We thank the other members of the Consortium for their assistance, namely, G. Sawyer and I. Jones-Parry [Marconi Space Systems (prime contractor)], D. Johnson (Marconi Research Centre), N. Hindley and J. Fullwood (Marconi Command and Control Systems), T. Pike and A. de Salabert (Messerschmitt-Bolkow-Blohm), J.-Y. Catros, T. Plassard, and I. Pottier (Thompson CSF), M. Barr and A. Wielogorski (Huntings Technical Services), and S. Dellapiane, S. Serpico, and G. Vernazza (University of Genoa). The NUTIS work was partly funded under NERC Contract F60/G6/12.

REFERENCES AND NOTES

(1) Tamura, H.; Yokoya, N. Image database systems: a survey. Pattern Recognit. **1984**, *17* (1), 29–43.

(2) Chang, S. K. Image information systems. *Proc. IEEE* **1985**, *73* (4), 754–64.

(3) Nagy, G. Image database. *Image Vision Comput.* **1985**, 13 (3), 111–6.

(4) Levine, M.; Shaheen, S. A. A modular computer vision system for picture segmentation and interpretation. *IEEE Trans. Pattern Anal. Mach. Intell.* **1981**, *3* (5).

(5) McKeown, D. M. The role of artificial intelligence to the integration of remotely sensed data with geographic information systems. *IEEE Trans. Geosci. Remote Sensing* **1987**, *25* (3), 330–48.

(6) Wu, I. K.; Cheng, D. S.; Wang, W. T. Model-based remotely-sensed image interpretation. *Int. J. Remote Sensing* **1988**, *9* (8), 1347–56.

(7) Nicolin, B.; Gabler, R. A knowledge-based system for the analysis of aerial images. *IEEE Trans. Geosci. Remote Sensing* **1987**, *25* (3), 317–29.

(8) Toriwaki, J.; Hasegawa, J.; Fudumura, T.; Takagi, Y. Pictorial information retrieval of chest X-ray image database using pattern recognition techniques. *Proc. Medinfo'80* **1980**; pp 1116–9.

(9) Corr, D. G.; Tailor, A. M.; Cross, A.; Hogg, D. C.; Lawrence, D. H.; Mason, D. C.; Petrou, M. Progress in automatic analysis of multitemporal remotely-sensed data. *Int. J. Remote Sensing*, **1989**, *10* (7), 1175–95.

(10) McKeown, D. M. MAPS: the organization of a spatial database system using imagery, terrain and map data. Carnegie-Mellon University: Pittsburgh, PA, 1983; Rep. CMU-CS-83-136.

(11) Rye, A. J.; Oddy, C. J.; Johnson, D. G.; Bishop, M.; Jones-Parry, I.; de Salabert, A.; Mason, D. C.; Bell, S. B. M.; Wielogorski, A.; Catros, J.-Y.; Plassard, T.; Serpico, S.; Hindley, N. MuSIP—Multisensor image processing. *Proceedings of the ESPRIT 1990 Conference*, 1990; in press.

(12) Stonebraker, M.; Rowe, L. A. The design of POSTGRES. *Proc. ACM SIGMOD* **1986**, 340–4.

(13) Herring, J. R. TIGRIS: Topologically Integrated Geographic Information System. *Auto Carto 8 Proceedings*, Baltimore, MD 1987; pp 282–91.

(14) Oddy, C. J. Picture Understanding Database—PUD system specification. Marconi Research Centre Report PALS/WN/108; 1988.

(15) Cruse, D.; Oddy, C. J.; Wright, A. A segmented image data base (SID) for image analysis. *Proc. IEEE 7th Int Conf. Pattern Recognit.* July 30–August 2, Montreal, 1984; pp 493–6.

(16) Haralick, R. M.; Minden, G. KANDIDATS: an interactive image processing system. *Comput. Graphics Image Process.* **1978**, *8*, 1–15.

(17) Rosenfeld, A.; Kak, A. C. *Digital picture processing.* Academic Press: New York, 1982.

(18) Merrill, R. D. Representations of contours and regions for efficient computer search. *Commun. ACM* **1973**, *16* (2), 69–82.

(19) Piper, J.; Rutovitz, D. Data structures for image processing in a C language and Unix environment. *Pattern Recognit. Lett.* **1985**, *3*, 119–29.

(20) Haralick, R. M. A spatial data structure for geographic information systems. In *Map Data Processings*; Eds.; Freeman, H., Pieroni, G. G., Academic Press: New York, 1980.

(21) Shapiro, I. G. Design of a spatial information system. *Ibid.*

(22) Shapiro, I. G.; Haralick, R. M. A spatial data structure. *Geoprocessing* **1980**, *1*, 313–97.

(23) Burrough, P. A. Principles of Geographical Information Systems for Land Resources Assessment. Clarendon Press: Oxford, 1986.

(24) Oddy, C. J.; Rye, A. J.; Tavendale, R. D. Software system design for general purpose image analysis. *GEC J. Res.* **1983**, *1* (1), 48–58.

(25) Petkovic, D.; Mohiuddin, K. Combining component features from multiple image frames. *IEEE Computer Society Workshop on Computer Architecture for Pattern Analysis and Database Management* **1985**, 169–74.

Technical Data Interchange Using Tabular Formats

PHILIP M. SARGENT

Engineering Department, Cambridge University, Cambridge CB2 1PZ, England

Received January 3, 1991

Formats and protocols used to exchange data between materials property databases are an enabling technology necessary for integration of materials information with computer-aided engineering. However, it is also necessary to ensure that the format is *at least as capable* of expressing "associativities" between data as are many of the participating databses. This paper shows that an extension to a "flat-file" or "tabular" data format can cope with arbitrary complexity in the associativity between the data items in a dataset (i.e., between the numeric data and the "metadata" describing the experimental conditions). The extension is simply to use several distinct tables to jointly hold the data: a *relational* tabular interchange format. Materials data, because of their heterarchically organized definitions and open-ended nature, present particularly difficult problems for data interchange. A solution to some of the problems is presented here, and it is expected that it may also be of use in other technical data transfer applications.

INTRODUCTION

A well-known commercial database format (dBase, by Ashton-Tate Inc.) is already being used by many materials properties database managers to upload test data into their databases. The main disadvantage of this format is that it is defined in binary. A plain-text version of this format is presented in this paper and maintains two-way "intertranslatability" in that no information is lost in conversion in either direction. It is called the Cambridge Tabular Data Interchange Format (CTDIF).

The aims of this format are to be

1. The simplest useful format
2. A flat-file tablelike representation
3. Convertible to and from dBase III *.dbf files
4. Plain-text to aid editing and word processing
5. Possible to include as an external reference in an SGML document
6. Compatible with being made conformant to MIL-STD-1840A
7. Designed for machine readability, not just a way to represent tables of data for presentation to people
8. Extensible to arbitrary complexity

In the past, the *associativity* and *meanings* of names and terms in interchange formats have been confused (a glossary of terms used in this paper is found in Table I). By separating these two aspects, this paper attempts to show that associativity

Table I: Glossary of Terms

associativity	The degree to which complex relationhips or associations between different items of information can be expressed, e.g., the associations between the items "temperature", "98.4", "Fahrenheit", and "human homeostasis"
compound document	A document which is computerized and which has different parts held in several different computer files, often containing different types of information such as text, graphics, numerical data, and digitized sound
definition file	A file in CTDIF-2 or CTDIF+2 format which contains the filenames of the CTDIF-1 or CTDIF+1 files which it thereby defines as a coherent group of files to be interpreted as a single set of information which has been broken up by a process of normalization (*qv*)
dictionary	A list of terms with a definition for each
ECMA	European Computer Manufacturer's Association
EDI	Electronic Data Interchange; This acronym is usually used only for business data
IGES	Initial Graphics Exchange Standard (U.S.) now available in version 4.0
MDI	Materials Data Interchange
metafield name	An agreed, standardized name with some specified meaning within a defined technical community, e.g.; a fieldname **#Validity#** which meant that each tuple would contain a string of agreed code describing some quality indicator of the data in that tuple
normalization	A technical sequence of operations that converts a single table of data into multiple tables without losing any information. It is usually done to reduce redundancy and to make it easier to enforce data integrity (see definition file)
ODA	Office Document Architecture (ISO 8613), effectively the same as ODIF; see SDIF
ODIF	Office Document Interchange Format (ECMA 101), see ODA and SDIF
SDIF	SGML Document Interchange Format, an "envelope" of information written in ASN.1 (CCITT standard language X.409) that makes it possible to send an SGML document over OSI standard networks
SGML	Standard Generalized Markup Language
STEP	Standard for the Exchange of Product Data, ISO Draft Proposed Standard 10303
thesaurus	A list of terms with information describing how each term is related to some other terms, e.g., narrower meaning, similar meaning, opposite meaning

is a technically difficult but eventually completely solvable problem—using current database technology.[1] By using a tabular format model, many of the problems of exchanging metadata with complex structures are reduced to naming problems only—a much simpler situation.

MATERIALS DATA INTERCHANGE FORMATS

A simple data interchange format is presented below. This is designed to offer the same facilities as dBase-III "flat-file" database files. The typical use envisaged is to take tables of test data results produced by an automatic data acquisition system and manually editing them so that they can be transferred to another organization for storage or evaluation. The format is entirely at OSI level 7, "application" level, and depends upon reliable end-to-end communication set up by lower levels.

The exchange format here is purely syntactical. No fieldnames for materials or properties are defined as it is assumed that users will agree to refer to some independently collated thesaurus of technical terms. The only exceptions to this are a few "metafield names" describing catalogue structures. It is also demonstrated how to make the tabular format compatible with other data exchange standardization activities. The formats described here are more fully documented in a Cambridge University Engineering Department technical report.[2]

Simplicity and User-Editing. The requirement that the format be simple for users to edit by hand has several implications. First, there may be no arbitrary "counts" such as numbers of records or lengths of strings to be typed in. It is awkward to have to re-edit a number at the beginning of a file whenever a spelling mistake is corrected near the end. The user should not have to make unnecessary decisions about the data when they are not relevant to his own work. Thus the length of text fields or the number of decimal places required for numerics should not be concerns for the user.

There should be no artificial and arbitrary limits on what can be typed; anything that obviously makes sense should be allowed. This is an ideal, but the proposed format approaches it more than most. Software is quite capable of counting numbers of fields, of distinguishing between strings and numbers, of measuring the length of the longest string, and of recognizing a wide variety of number formats, and it makes no sense to burden the user with such tasks.

dBase Files. The "dBase" file format is a de facto industry standard for the communication of database files between DOS-PCs. Nearly all other database software products for PCs (e.g., Borland's Reflex v2 and Paradox v3, Lotus' 1-2-3 v3, Quadbase's dQuery) support translation to and from this format, and Ashton-Tate have published the specification.[2] The same file format is thus supported on more types of computers than will actually run dBase itself.

The format describes a single "flat-file", best thought of as a two-dimensional table of data where the columns are named with "fieldnames" and the rows, or "tuples", are not. There are a number of restrictions on the range of numbers permitted, the lengths of strings, the number of fields (columns), and the total size of any one file, but they are mostly not too onerous if they are managed in software, which is essential, because the format, being in binary, cannot be hand-edited.

CAMBRIDGE TABULAR DATA INTERCHANGE FORMAT

The plain-text translation of dBase data files is called the Cambridge Tabular Data Interchange Format (CTDIF). This format is best understood by first studying an example. In the following file, the keywords are IMPLEMENTATION, NAME, FIELDLIST, ENDFIELDS, CTDIF-1, and FIDTC-1. Strings are enclosed in double quotes only if they contain spaces of other separator characters. Only two data types are used: strings and numerics. Numerics are recognized whether they appear as integers or decimals, with or without an exponent. Separators can be spaces, linefeeds, tabs, or commas in any combination. Multiple separators are not significant.

```
CTDIF-1 0.1
implementation "PMS dBase Convertor v0.1 21-July-1989"
name NIMONICB updated 89/7/21
fieldlist "sample no" weight length strength_MPa elongation_to_fracture
endfields
&&1-fred 3 5.0e-4 200.3 0.23
&&2BA 3.2 1e-3 205.2    0.235
"&&3Z ++" 3.333 1e-3 205.3   0.236
FIDTC-1
```

It can be seen that the data consists of a number of "tuples" (rows or records), where each tuple contains one value for each fieldname. The word "tuple" is used because the separators are not necessarily the spaces and linefeeds shown above which put one tuple on each row.

The IMPLEMENTATION string in the example describes the software that was used to prepare the datafile. This might be, for example, a person's name and a word-processor package. The NAME field refers to the original name of the dBase file, and the UPDATED field refers to last date the data were edited or typed. The names of the fieldnames are listed between the keywords FIELDLIST and ENDFIELDS before the data themselves appear.

Automatic Recognition of Types. The numeric and string types can be automatically recognized by the values that appear, so that &&1-fred is clearly a string and 5.0e-4 is clearly a number. This detection routine must examine all values of a field before concluding which type it is. If the user requires strings which consist only of digits then quotes should be used, as in "007". There is a slight danger that a single typing error such as O (oh) for 0 (zero) could cause an entire set of numbers to be classed as strings, but the translation software should be alert to this possibility and produced appropriate warning messages.[2] Thus the format is "strongly typed" but it does not require any type declarations and thus is like the programming language ML but unlike Pascal.

Restrictions. We must apply a number of restrictions if free interconversion with dBase files is to be maintained. dBase does not permit NULL values for numeric (or date) fields. This lack of NULL is a severe disadvantage for materials property data where absence of values is very common. Ways of providing this facility are described in the detailed format definition.[2]

Some of the restrictions are awkward and unpleasant, but rather than extend them immediately it makes sense to define a base format (this one) and a separate, upwardly compatible extended format which removes the restrictions. This is because there is a great deal to be gained by having a dBase-convertible format.

SGML AND CALS CONFORMANT EXAMPLES

Any data, in any format, can be referred to from a Standard Generalized Markup Language (SGML) document by the name of the file in which it is stored.[3] This is true for images, vector drawings, recorded speech, or music, anything. If there is a formatting program available that can produce printed output from such a data file, for example, a translator that can produce written music from a recording, then by referring to this program from within an SGML document it is possible to include the data so that they appear in the right place when the SGML document is printed. The tabular data format proposed here presents no problems when used in conjunction with SGML and in this it is also compatible with ODA (Office Document Architecture) and ODIF (Office Document Interchange Format) by encoding the SGML using SDIF (SGML Document Interchange Format).[1]

CALS Conformancy. The U.S. Department of Defense program of Computer-Aided Acquisition and Logistics Support (CALS) currently does not specify a format to be used for tabular data. IGES (Initial Graphics Exchange Specification) formats are required for engineering drawings and vector graphics (later migrating to STEP [STandard for the Exchange of Product data, ISO DP10303] as it becomes available), bitmap, raster and tiled-raster formats are defined for images, but only the very general military standard 1840A is defined for numeric data.[4]

CALS uses SGML format text documents to link together compound "documents" consisting of information recorded in any of the other formats using part of the SGML facilities described above. Thus while CTDIF could be considered to be CALS conformant purely because it can be used with SGML, this interpretation may not be accepted by other users of CALS standards and so conformance with MIL-STD-1840A must be considered. This military standard requires that files of data conforming to it be expressed in ASCII characters. CTDIF format can be reformatted in any layout without changing its meaning, so CTDIF files can be made conformant to MIL-STD-1840A.[2]

MULTITABULAR (RELATIONAL) FORMAT

The simple, single tabular exchange format has many drawbacks even without considering restrictions imposed by dBase compatibility. These are

1. Repeated data is repeated in the transmission
2. Single-point data requires a whole file to itself
3. The available associativity is only very simple[5]

These can be alleviated by using multiple data files to describe the same set of data, a technique known as "normalization". If all the data are sent (or stored) in a single table as defined above, then they are "simply normalized" or in "1st normal form", and further normalization removes redundancy without losing information. The principle involved is very simple: any one item of information should only be represented once. Howe gives a "cookbook" approach[6] and describes what to do in order to achieve better normalization (and gives many examples), and Date[7] describes clearly and concisely why the method works. Ullman[8] gives formal mathematical proofs. A good bibliography is provided in the Proceedings of the Schluchsee Workshop.[9]

Cambridge Multitabular Format. The simplest extension to the simple tabular format is an extra *definition file* added to a *set* of CTDIF-1 files encoded according to the simple tabular form. To distinguish this extension from the simple case, the format of the new definition file is called CTDIF-2 and contains a list of the filenames which refer to simple CTDIF-1 files. A further step would be to add at the end of this definition file (before the CTDIF-2 terminator) the *contents* of all the simple files, complete with CTDIF-1 headers and terminators, rather than just their names. This has the advantage of neatness and conceptually provides precisely the same descriptive power. Here is an example of what the definition file might look like. Note the descriptive comment on line 4:

```
CTDIF-2 0.1
implmentation "AMCENS Ctdif&&Z version 0.2 01/4/1989"
name RRNIM
"This is a whole bunch of stuff bought from RR"
filelist c:\NIMONICA.c-1 c:\rr\NIMONICB.c-1 d:\new\NIMONICC.c-1
NIMONICD.c-1 NIMONICE.c-1 endfiles
FIDTC-2
```

Unfortunately, in practice, effective normalization becomes increasingly difficult for the user as the associativity increases in complexity, and there are many traps for the unwary.[6,7,10]

EXTENDED FORMATS

The single-table and multiple-table CTDIF-1 and CTDIF-2 formats are files or collections of files which are dBase compatible, but this will not always be adequate. The *extended* CTDIF+1 is, like CTDIF−1, a single-table format, but most of the restrictions associated with dBase compatibility are removed.[2]

The extensions to the single-table format can be carried through to collections of these tables as in the multitabular format CTDIF+2. This gives new freedoms, but also new restrictions.

CODATA GUIDELINES

After the CODATA-sponsored Schluchsee meeting in 1985, an informal working group was established to devise a materials data interchange format, and a set of guidelines was produced. The format envisaged by CODATA must be understandable by scientists or engineers who are only casual users of computers and it must support the following facilities:

1. Numeric data with uncertainties
2. Graphs and their metadata

3. Tables and their metadata
4. Data functions (mathematical expressions)
5. Supporting text information
6. Other, non-Latin alphabets and extensions
7. Common mathematical symbols as extended ASCII codes
8. Units

It should not however attempt to support diagrams, pictures, and the full text of articles. In addition, the format itself should have the following properties:

9. A route to future evolution without invalidating past versions
10. Easy recognition of a "unit" of data
11. Definition of new variables within the datafile
12 Easy identification of the beginning and end of the data
13. Identification of the sender
14. Identification of the intended recipient
15. Provision for naming a "unit" of data (a datafile)
16. Provision for the data and time of the request for the information
17. Location and delimiting of any "structural element" within a datafile

It can be seen that the Cambridge multitabular format CTDIF-2 handles most of these. The uncertainties of numerics and the specification of the structures of graphs can be modeled using multiple tables. This must be so since the full power of the relational model of data is available. The "definition of new variables" is the only defined behavior for CTDIF: all fieldnames are declared but their *meanings* are *not* defined. Mathematical expressions can only be represented as strings, which is probably adequate.

Latin and Greek alphabets are independent of the definition of CTDIF. The ISO has defined "octet" (byte) codes for several non-Latin alphabets, including Japanese, which requires 2-byte codes, and CTDIF could be used directly once translations of the keywords such as FIELDNAMES have been defined.

CONCLUSIONS

Simple tabular and multitabular (relational), dBase-compatible and extended formats all have their uses, and it is inevitable that if only *one* of these is adopted by a standards-making body some user-communities would then develop others. It makes sense to forestall such a divergence of standards by designing a system where all types can coexist and where there are defined methods for transforming from one to the other. The work referred to in this paper defines Version 1.0 of CTDIF–1 and Version 1.0 of CTDIF–2 (dBase-compatible). It gives suggestions for the extended formats CTDIF+1 and CTDIF+2, but further thought is required to arrive at a sensible and workable set of extensions.

The systems described here, however, contain no mechanism to handle *meanings* or *interpretation of terms*. Therefore the CTDIF formats will only be useful to a data-sharing community if that community *also* defines a common fieldname dictionary *with definitions* to ensure that meanings are also communicated as intended.

REFERENCES AND NOTES

(1) Sargent, P. M. Definition Study for the Establishment of Demonstrator Projects in Materials Data Interchange. Contract No. 320111 for CEC JRC Petten; June 1988.

(2) Sargent, P. M. A Tabular Materials Data Interchange Format. Cambridge University Engineering Department Technical Report CUED/C-MATLS/TR.162; Aug 1989.

(3) Bryan, M. *SGML: An Author's Guide to the Standard Generalized Markup Language*; Addison-Wesley: Reading, MA, 1988; ISBN 0-201-17535-5.

(4) MIL-STD-1840A Military Standard: Automated Interchange of Technical Information. Dec 22, 1987. Also: Change Notice, Dec 20, 1988. U.S. Department of Defense; Section 5.1.4.2 p 19 and 5.1.4.9 p 25.

(5) Sargent, P. M. Use of Abstraction in Creating Data Dictionaries for Materials Databanks. ASTM International Symposium on Computerization of Materials Data, Orlando, FL, Nov 1989.

(6) Howe, D. R. *Data Analysis for Database Design. Part 2. Normalization. Part 3. Entity-Relationship Modeling.* E. Arnold Publ. Ltd.: London, 1985; ISBN 0-7131-3481-X.

(7) Date, C. J. *An Introduction to Database Systems*, 4th ed.; Addison-Wesley: Reading, MA; Vol. I, 1983; Vol. II, 1988; ISBN 0-201-14474-3.

(8) Ullman, J. D. Dependences and Lossless Decomposition. *Principles of Database Systems*, 2nd ed.; Pittman: London, Chapter 7, 1982; ISBN 0-914894-36-6.

(9) Westbrook, J. H., Behrens, H., Dathe, G., Iwata, S., Eds. *Material Data Systems for Engineering*; Proceedings of the CODATA Workshop, Schluchsee, Germany, 1985; ISBN 3-88127-100-7.

(10) Kent, W. *Data and Reality: Basic Assumptions in Data Processing Reconsidered*; Elsevier North-Holland: New York, 1978; ISBN 0-444-85187-9.