LANGUAGE REQUIREMENTS FOR UNDERSTANDING RETRIEVED DATA

Fred C. Billingsley
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, USA

INTRODUCTION

A primary focus of information systems activities is to provide the techniques and tools to enable data users to locate, acquire, and utilize data in the continuing search for understanding. This requires the interfacing of multiple data centers and data archives in order to provide access to their information sets and supporting tools and services.

The standardization being planned and implemented for space information systems stresses interoperability - the attempt to make the interfacing "seamless". When there is a requirement to obtain information from another location or an archive, there is a parallel need to transmit a complete description. This description is the function of a Data Interchange Language (DIL). The DIL should be powerful enough to enable parsing of the transported information, whether transported via electronic or recorded media. The ultimate objective is to minimize the need for manual intervention in the transfer process.

The objective of this overview is to describe the method of facilitating this transfer between dissimilar computers, using the Transfer Syntax Description Notation (TSDN). This notation is being developed as a generic capability to allow transfer of discipline-dependent information in a discipline-independent and medium-independent manner.

THE GENERAL DATA MODEL

"Data" is a broad term. It has been convenient to divide the concept into smaller parts, so they may be considered separately. This has historically been done in the industry using the terms Volume, File, Record, Elements or Data Fields, byte, bit, representing progressively smaller entities. These are usually considered as nested.

We will continue to use these terms where appropriate. We also expect to make maximum use of available commercial capabilities to avoid the need of inventing the computer industry over again. Specifically, and for example, we will use the conventional definitions for byte (or Octet), and bit. We also expect to find useful the industry concepts of, and services relating to, records and files.

The TSDN has been developed with a logical model of organization of the data it is describing as:

```
Volume        ::=  +( Files )
File          ::=  +( Records )
Record        ::=  +( Data Fields )
Data Field    ::=  Compound | Simple
```

```
Compound Field   ::=  +( Compound | Simple )
Simple Field     ::=  Octetstring | Bitstring
```

The notation ::= means "is composed of"
+(...) indicates "consists of one or more ...".
| indicates alternate

Within any closed endeavor, several things are required to be defined for each entity and aggregation:

An instance identification - its name.
How to recognize or bound it.
How to parse it - its format or syntax.
Relationships among its parts and to other entities.

As long as the endeavor is closed, these definitions are arbitrary, but all of them must be stated, either directly or as analogs to terms used outside of the endeavor. When the TSDN module is transferred or makes use of commercial (i.e., outside) capabilities, these terms must be mapped into a language understood by the outside facilities. "Record" needs a special comment. A number of external systems and languages do not recognize the concept of record. Therefore all references to "record" will mean logical data record, and definitions are needed for use within the endeavor. Particular care must be taken that the internal protocols do not interfere with the external ones.

REQUIREMENTS AND ATTRIBUTES FOR A DIL

A number of requirements have been suggested for the data language (DIL) to be used to write the description. Among them are:

Ability to be used either separated from or attached to the data file, to describe already-standardized and fixed formats.
Ability to name and describe data primitives, including machine binary, non octet-aligned binary fields, and the various character mode numerical representations of ISO 6093.
Ability to name and describe commutated and other structured data.
Ability to describe logical relationships between data objects in a record.
Ability to describe logical relationships between data records in a file.
Ability to describe logical relationships between data files in a volume.
Ability to reference data at various levels of abstraction.

In addition, there are several attributes necessary for acceptance:

Must be application-independent.
Must be medium-independent.
Must be machine-independent.
Must be conceptually simple and self-documenting.
Must be human readable after minimum training.
Must be usable
between diverse DBMSs;

between programming languages;
by non-programmers.
Must provide reference to externally-defined syntaxes.
Must be flexible, contractable, and expandable to allow the DDR to be built with a minimum of overhead.
Must be able to be written with standard word processors.

These will not be belabored here. Suffice it to say, that the TSDN has been designed to try to meet these requirements and attributes.

Specifically, the TSDN permits the sender to describe the transferred information and to send this description separately or as an integral part of the transfer file. It permits the description of both character and bit field information in fixed- (without delimiters) or variable-width (delimited) fields or subfields. It further permits the identification of fields and subfields by arbitrarily long names and labels which serve to give meaning to the data. In addition, it provides for the definition and labeling of complex structures and commutated data.

The Goal

The intent is to supply the data product user with the conceptual or logical description of the data as well as the format and representation of the data. This information may be packaged with the data such that a suite of standard software at the user's installation can transform the data to conform to his machine architecture and can present standard views for applications.

Why is Something New Needed ?

Type declarations in programming languages define the element formats as they occur within the closed system of the hardware/operating system housing the analysis program. Inasmuch as these declarations are different from language to language, and do not pretend to describe the formats as the data are transferred, a common notation for describing the data during interchange is required.

The Problem

Analysts may be expected to work in an environment in which there is a need to retrieve data from various files, either local or in archives, generated recently or decades ago, and to present that data to the analysis system in a usable, verified form. The files may be expected to be in various formats, using various numerical representations. The applications requiring the data may be expected to be in any one of a number of languages. For the system to be useful, a simple and consistent interface to the user programs is needed. Such an interface should be able to respond to a request from a user analysis program containing elements such as:

retrieve <a data element, by name, as found in the descriptive file>
from <archive and filename of the file containing the data element>
name <new name as required by the analysis program>
format <format in the user computer/language application>

Format Description - The TSDN

The philosophy behind the TSDN construction is that a data transmission consists of a series of bytes. These bytes have no inherent meaning nor interrelationships, requiring that everything be defined. The user may define keywords (Labels) for the data fields of the user modules. These fields may be described and retrieved by using the labels as keywords.

These definitions are carried in a series of statements, or SEGMENTS. This convention defines the use of these descriptive Segments to describe the structure of a file and each data record in the file.

Each TSDN segment has the same basic structure:

Segment Tag Tag Delimiter Segment Value Segment Delimiter

The structure of the segment value field varies, depending on the use of the segment.

The TSDN may be described as Keyword-driven, where the TSDN keywords are segment tags. This allows the building of a TSDN Module from a relatively small group of specification-defined segments plus user-defined segments. This approach allows a standard, recognizable, group of segment tags to be specified for the TSDN, from which a given instance may be assembled. Only those segments needed in a given instance need be used.

Data objects may be simple (primitive or compound) or aggregate (file, record, structure). There are only three fundamental data types, from which others may be constructed:

Primitive - named, undivided
Compound - named, a sequence of unnamed components
Aggregate - named, components named individually, with relationships.

Only one TSDN segment type (OBJECT) is required to describe all aggregate objects, at whatever aggregation level. An OBJECT segment is used to describe the logical structure of each aggregate object, while a TYPE segment is used to describe the concrete syntax (format) of each primitive or compound data field. Other segments are defined for use in describing other aspects of the data.

An Example

Let us take as an example, the personnel record illustrated in Appendix E of ISO 8824 (ASN.1).

Name:	John P Smith
Title:	Director
Employee Number:	51
Date of Hire:	17 September 1971
Name of Spouse:	Mary T Smith
Number of Children:	2

Child Information
Name: Ralph T Smith
Date of Birth: 11 November 1957

Child Information
Name: Susan B Jones
Date of Birth: 17 July 1959

We will visualize a data record in which the data fields are separated by commas and terminated with an exclamation point, thus:

John,P,Smith,Director,51,19710917,Mary,T,Smith,2,Ralph,T,Smith,19571111,
Susan,B,Jones,19590717!

The generic format description must allow for a variable number of children, which is made specific for each employee. The Object segment describing this record would be (TSDN is case-insensitive):

OBJECT = personnelRecord,SEQUENCE,(name , title , number , dateOfHire , nameOfSpouse , numberOfChildren , numberOfChildren(childInformation)) ;

OBJECT = name , SEQUENCE , (givenName , initial , familyName) ;
OBJECT = nameOfSpouse , SEQUENCE , (givenName , initial , familyName) ;
OBJECT = childInformation , SEQUENCE , (name , dateOfBirth) ;

The interpretation is straightforward except perhaps for the numberOfChildren(childInformation). The TSDN syntax here uses a numeric variable (numberOfChildren) adjacent to an opening parenthesis to indicate a quantity-of.

The components of name are individually identified, and thus become data fields, requiring the comma separators. This allows searching on familyName, for example.

The complete File would be a set of these records:

OBJECT = PersonnelFile , SET , +(PersonnelRecord) ;

If this file is defined as a system-available file, this identification may be made by:

FILEID = PersonnelFile , NONE , <fileName> ;

Here the external file name is the <fileName>, the NONE indicates no special protocols or conditions, and PersonnelFile is the TSDN-internal object name for the file.

It now remains to define the formats of each individual data field. Each format definition consists of a syntax code plus a length or terminator. Some of the syntax codes defined in the specification are:

B	primitive bit data field; numerical binary unsigned uncomplemented
A	primitive (undefined) character (alphanumeric) data field
I	integer character string (ISO 6093 NR1; 8824 char int)
D	decimal character string (ISO 6093 NR2; 8824 char dec)

```
  E    exponential character string        (ISO 6093 NR3; 8824 char real)
  F    floating exponential, binary        (ANSI/IEEE 754 "Floating Real")
  R    floating exponential, binary        (ASN.1, ISO 8824 "Real")
  Z    character positions with no format explicitly defined
```

The TYPE segments which define the concrete transfer syntax (formats) of the individual data fields will be:

```
TYPE = givenName,(A,)          ;
TYPE = initial,(A,)            ;  -- Allows multiple initial
TYPE = familyName,(A,)         ;
TYPE = dateOfBirth,(I,)        ;  -- Semantics is YYYYMMDD
TYPE = title,(A,)              ;
TYPE = number,(I,)             ;  -- Allows multi-digit employee number
TYPE = dateOfHire,(I,)         ;  -- Semantics is YYYYMMDD
TYPE = numberOfChildren,(I,)   ;
```

All of the fields are variable-length, terminated with the comma delimiters. The "!" record delimiter is pre-defined within TSDN as the default value. All that now remains is to complete the mechanics of the TSDN syntax:

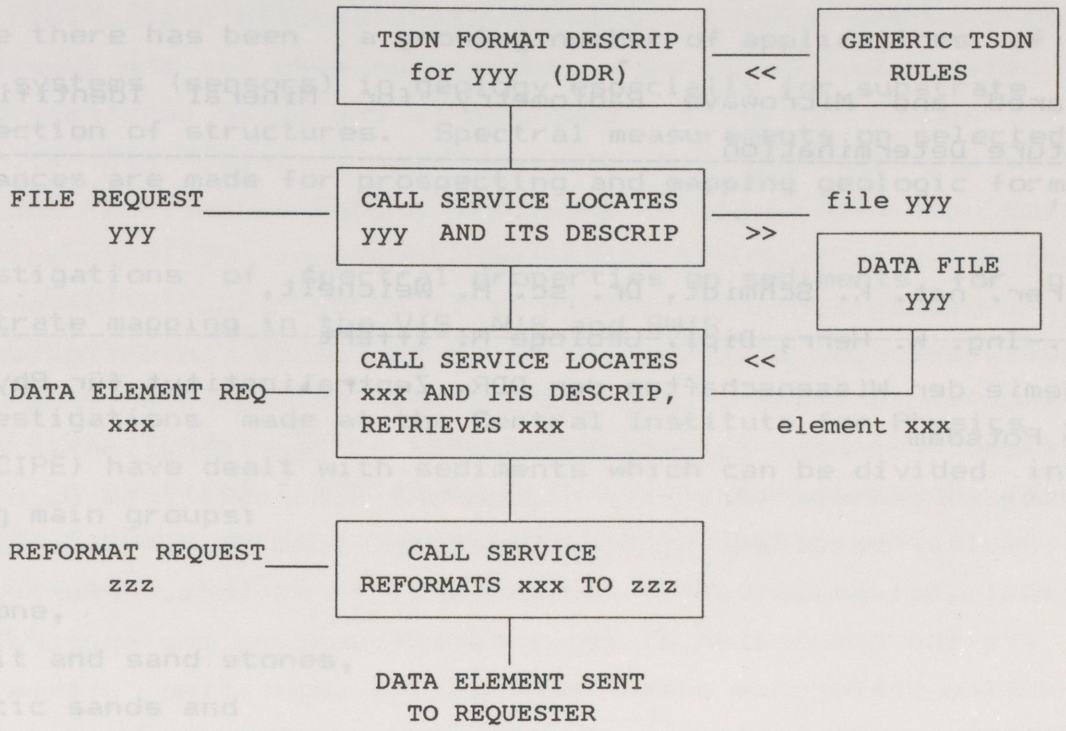
```
MCAI = TSDN,1990-09 ;                          -- Identifies TSDN date of issue
UID = ASN.1 Personnel Record Example ; -- User-defined identification
TYPE Segments
OBJECT Segments                        -- Segments as described above
FILEID Segment
END.                                   -- Completes the Detail Section

                                       -- If there were data records att-
                                          ached, they would be here.

eof                                    -- Completes the TSDN Module
```

Data Conversion

In a heterogeneous system containg diverse computers and programming languages, it is almost certain that the data as retrieved from a remote system or archive will be in different format than that required by the recipient. By using the description transmitted with the data (or on hand apriori) conversion routines may be invoked to convert the representations to the required form, with a minimum of intervention by the recipient.

This is the point at which the retrieval request is answered. A service is anticipated, and being prototyped, which will read a TSDN description and deliver to the application a properly-formatted data entity. (This service is not part of TSDN per se, and is given here only as an illustration of a method of use of TSDN.)

Consider a call such as: Select xxx from yyy in format zzz.



MODEL FOR DATA ELEMENT RETRIEVAL AND CONVERSION

TSDN Status

The TSDN is being developed at the Jet Propulsion Laboratory, Californua Institute of Technology, under the auspicies of the Consultative Committee on Space Data Systems, under a contract with the National Aeronautics and Space Administration. It has been accepted by ANSI as a new work item; it is hoped that it will be imminently ready for public review. Further information is available from the authors, and comments will be appreciated.