REPORT TO X3 ON DATA DEFINITION LANGUAGES

J.A. Gosden
Chairman, X3 Ad Hoc Committee on DDL

[Editor's Note: The following report consists of John Gosden's recommendations of the action X3 should take in the area of Data Descriptive Languages. The report has, at this date, been accepted by X3 (which means they have acknowledged receipt of it). It has not been approved nor has it been rejected. In addition, the report is Gosden's, in that it represents his view of the committee findings and feelings. It has, however, not been officially approved (nor rejected) by the Ad Hoc Committee as a body. The report is published here to make it available to the public, to permit them to read it, and perhaps comment on it as they see fit. Whatever action is taken by X3, the absence or other role of standards bodies in the area of the use and transfer of data will be significant, and X3's action will be based, to some extent, upon the contents of this report.]

INTRODUCTION

The following report was prepared after considerable USASI and other activity which is outlined herein. This report synthesizes my version of conclusions reached at a number of informal, but intensive, discussions held at the SJCC 1969 with D. Hatfield, T.W. Olle, H. Tompkins, C. Mooers, and H. Morse. It has been reviewed by them and I have incorporated their major comments. It is now being forwarded to X3 and I request that BEMA distribute the copies to the Ad Hoc Committee. I also request that permission be given for this report to be published in the ACM SICFIDET bulletin, fdt. We do not think a further meeting of the Ad Hoc Committee is necessary. I recommend that X3 discharge this Ad Hoc Committee.

A. The Problem

The charge to this Ad Hoc Committee by X3 is attached in Appendix A.

B. Activity

The first meeting had before it several background documents. There was general discussion. Three subgroups were formed to prepare statements of:

1. The need
2. The status today
3. The future problem.

Subgroup 1, chaired by Ethel Marden, produced a Statement of Need (Appendix B). Subgroup 2, Sable and Olle, sent out a broadcast letter and collected a variety of responses which have not been processed. (The material is in repository.) Subgroup 3, chaired by Morse, became hardly distinguishable from the ACM Special Interest Committee on File Description and Translation (SICFIDET, which came into existence during our study and is chaired by Hatfield), and prepared a draft statement of the technical problem (Appendix C). SICFIDET discussed DDL's at the SJCC 1969 and later CODASYL discussed the DDL problem at its Tenth Anniversary Conference, and now the CODASYL Systems Committee, chaired by Olle, has a strong interest in doing development work on DDL's.

C. Basic Jargon (Glossary)

In this paper we define *language* as "a system of communication", be it tables, text, codes, pointers, or whatever. Olle points out that something can be *described* as *terrible* and thus recommended that, henceforth, we talk about Data *Definition* Languages (DDL's). We endorse his recommendation and have used it throughout.

D. Technical Background – What is a DDL?

There are three basic varieties of languages that we need to discuss:*

DDL-I – informal language used by humans (in default of DDL-F's)

DDL-F – formal language used by humans (to communicate with machines or other humans, e.g., COBOL, FORTRAN)

DDL-M – machine language or hardware representation (used to communicate between machines or software processors).

There are three classes of DDL. They are distinguished by the topics which they discuss:

C-DDL (Conceptual) – in which we can express definitions of data structure concepts. This is needed to be able to define the domain of data structure capabilities of various systems.

L-DDL (Logical) – used to define *data structure* as the user sees it including *all the logical structure* (relations and properties) *and* as much of the *physical structure* as appropriate, e.g., when describing input data, it is necessary to include the field positions but it may not be necessary to describe the physical structure of data in a backing store. One example is the COBOL DATA DIVISION. These languages will be peculiar to various systems. Descriptions in these languages are human-readable versions of corresponding descriptions in P-DDL's.

P-DDL (Physical) – used to define data structure as the machine sees it, including *all the logical structure* and *all the physical structure*.

[N.B. The concept of complete data interchange (especially between different software systems) requires that physical data descriptions be complete and explicit. Thus we have, at the hardware level, potential procedure and data independence.]

---

*This discussion is based upon a presentation by T.W. Olle to ACM SICFIDET, 14 May 1969.

Table I shows some examples of types of data definition languages, examples of a particular class and variety.

TABLE I. Types of data definition languages.

CLASSES	VARIETIES		
	DDL-I	DDL-F	DDL-M
C-DDL (Conceptual)	"A" Technical English	Knuth, Vol. I	
L-DDL (Logical)		"B" COBOL Data Division	IAL
P-DDL (Physical)	MARC II(a)	"C"	"D" MARC II(b)

Olle discussed four DDL's, which he labelled A, B, C, and D. These are shown in Table I: A = C-DDL-I; B = L-DDL-F; C = P-DDL-F; D = DDL-M. The following notes are applicable to the Table:

1. Knuth, Vol. I [*1*] refers to the elegant formal *language* developed by Knuth to describe logical data structures.
2. IAL is an example of some representation a compiler must have internally to correspond to a POL's data description.
3. MARC II(a) is the language of the manual or standard document.
4. MARC II(b) is the language of the data definition tables and codes written in MARC tapes.

MARC II is DDL-developed by (and being used for bibliographic data originating from) the Library of Congress and is the basis for a proposed standard prepared by Z39. Henrietta Avram (Library of Congress) chairs the Z39 working group responsible and was a leader of the MARC development.

E. Status Report — Where are We (They)?

The situation is confused: We do not have a categorization of DDL's and people are often confusing different types of DDL and also confusing different levels of solution of the problem. Let us use an analogy and compare DDL's with POL's. A rough comparison is shown in Table II.

TABLE II. A comparison of DDL's with POL's.

LEVEL	POL	DDL
(i) Machine Level	Assembly code	Implicit in data and programs
(ii) First useful version in limited area	BØ B1	FFS NIPS – IDHS | FFT
(iii) Synthesis of many versions with wide applicability but limited in scope	COBOL : FORTRAN IV	MARC I MARC II P-DDL-I and P-DDL-M
(iv) As in (iii) but with wide scope	PL/I (?)	JAG – Goal P-DDL-F and P-DDL-M
(v) Compatibility	UNCOL (?)	UDDL

Thus we might say that MARC was a CODASYL for data definition, and much of the criticism of MARC II is due to some people having claimed (by analogy with 20/20 hindsight on POL's) that MARC II is not a suitable basis for evaluation to level (iv) and that we thus stand in danger of not getting to level (v) if there is a great proliferation of uncoordinated implementations. MARC II does not, in my opinion, handle many hierarchical levels well, nor does it allow much variety of data structures. In particular, because a nested structure is cumbersome when extended, a list-structured DDL is essential for growth. There is still no consensus whether a different approach is needed at level (iv) or whether an evolution to a hypothetical MARC III is a better answer. Clearly such differences have been largely due to the *level of view*. At this point Olle and I should note that we do not think that proposals to extend the data division of COBOL are a realistic solution to the problem.

The theorists say that *the solution* is many years away and that many DDL's will be required. They are talking about the total problem, level (v), whereas MARC, level (iii), is addressed to one small fraction of the total problem (but potentially huge in absolute terms of data traffic). JAG (a Joint Agreements Group which is composed of representatives from such professional groups as ACM, AIP, IEEE, Chemical Abstracts, NBS, and EI who have an interest in originating and exchanging technical bibliographic data) was shooting for level (iv) on a large fraction of the total traffic but still only a small fraction of the possible variety. SICFIDET has been discussing mostly level (v) (i.e., the total problem) or, at least, a large fraction of the possible variety, and rightly claiming that a single universal DDL (UDDL) is unfeasible and that many DDL's will be required. There are parallels in the POL community.

F. The "Escape" Clause and Registries

In spite of all these problems, we can "live at all levels". We can even pretend that we have one UDDL when we have many, as we do in the case of the "escape" characters that allow us to have many character codes within one character code.

We need a general rule to tell us where the DD (data definition) is located in a file. The first set of characters (x) is read by the level (v) translator controller to tell it which of the general purpose level (iii) and (iv) translators to enter, the next set of characters (y) is the DD in P-DDL-M (x). One member of the set of x (x') is an escape set which says that (y) is the identifier of an unparameterized subroutine built to translate a specific level (i) or (ii) file.

Thus, initially, we have many level (i) or (ii) individual file translators and slowly phase these out into level (iii) or (iv) general purpose translators as needed.

Of course all experienced *standardites* (and others) realize it is not as easy as it sounds.

G. What Do We Need To Standardize?

1. We will need to standardize the widely used P-DDL-M's in which DD messages (character set y's) are encoded and for which we wish to build level (ii)/(iii) translators.

2. We will need a registry to control identifiers for specific level (i) and (ii) translators (character set y). A registry for y may seem impossible at first, but can be decomposed into sections

   $$y_1\ y_2\ y_3\ y_4\ y_5\ \cdots\cdot$$

   where y_1 names to standards body (e.g., 5 = USASI), y_2 names major groupings (e.g., 1 = DoD, 2 = IBM, 3 = ····) who each maintain direct lists or further subregisters as necessary.

3. We will need a registry to control the character sets x to identify various DDL's. This can be hierarchical in the same way as 2 above.

4. We will need guidelines for translation between systems of different power, e.g., limitations on the use of capabilities in the source DDL if a translation with no information loss is required to a specific object DDL; and definitions of the loss to be expected if the guidelines cannot be followed.

5. We will need conventions for translation between systems of different styles where many alternatives are possible, e.g., conventions for mapping *tables* into *trees*, and careful definition of such structures.

H. What Should X3 Do?

1. Set up the national registry mechanisms and formats soon with a way to get ISO included in the top level of the hierarchy. Try to include Z39

with minimum disruption. Perhaps "x" should be in file labels.

2. Set up procedures to ensure other efforts such as Z39 liaison with X3.

3. After suitable X3 reflection, liaison with X3.4 and X3.8, and perhaps some more specific technical progress, create a counterpart to X3.4 (POL's) to handle DDL's (call it X3.20) and do not get DDL's mixed up with code values, item names, etc., as in the present X3.8. This is perhaps some 6 to 12 months away.

4. Encourage data and procedure independence by restricting X3.4 to POL's and encourage new POL's to be data independent. Let their L-DDL-F's be processed by X3.20.

5. Get X3.20 to try to resolve the nonproliferation problem in DDL's and standardize level (iii) DDL's.

6. Encourage CODASYL, Z39, JAG, and others to develop potential solutions to the nonproliferation problem, level (iv), and bring them to X3.20 for standardization.

7. Encourage SICFIDET to worry about the general theory, level (v), attack the translation problems between level (iv) solutions, and to be a forum for technical development and discussion.

8. Encourage CODASYL, SICFIDET, SIGPLAN, IFIP TC 2, to consider the problem of uncoupling data from the POL's of today.

APPENDIX A

Subject: Data Descriptive Languages

Reference: 7 October 1968 Ad Hoc Committee Report on Scope

X3/SAC moves that X3 accept the proposed scope of the Ad Hoc Committee on Data Descriptive Languages with the recommendation to the Ad Hoc Committee that since its singular task is to recommend to X3 what work should be done on data descriptive languages that it is presumptive to fix upon a firm definition of what a data descriptive language is at this time. The definition of what a data descriptive language is should follow from the committee's study of data structures among classes of users and the key parameters which must be specified in order to achieve information interchange of data. The critical parameters should be classified as part of the definition.

APPENDIX B

STATEMENT OF NEED FOR A DATA DESCRIPTIVE LANGUAGE

by Subgroup 1

Computer technology has evolved in the last fifteen years in a singularly *free* fashion, without regulation, without constraints of tradition, without the rigidity imposed by the continued use of obsolescent equipment, and without the restraints of conformity or standards. Systems designers and programmers, therefore, have been free to describe information to be manipulated by the computer in the manner best suited to the available equipment, the nature or structure of the information to be handled, the requirements of software, or the idiosyncracies or preferences of the innovators of the system.

The freedom heretofore employed in data description is now limiting the use of both data and programs and is making their interchange more difficult and costly. It has been estimated that the lack of an adequate data description is costing the Department of Defense millions of dollars annually because of the inability to exchange data effectively. General recognition of this lack, along with cognizance of other deficiences, undoubtedly provided motivation for the X3 Committee of USASI to set up an Ad Hoc Committee to consider a Data Descriptive Language. Some of the more compelling reasons for the development of a DDL are discussed in this Appendix.

Interchange of Information

In order to interchange data there is a need to describe data formats. The description must include as a minimum (1) the identification of the data elements in the record or message, (2) the location of the data elements in the character string of the record or message, and (3) the form of representation, i.e., names, abbreviations, codes, or numeric expressions, used in the identification of the data elements, together with their specific values (data items). As information systems become more sophisticated, there is also a need for the description of relationships among the data elements, i.e., primary, attributes, nesting, etc. Data are frequently tied to specific programs in that the procedures of the programs are dependent upon derived values of the data and thus their transfer requires great effort, usually including the development of special translation programs or complete reprogramming. The file/record or file/record/index level of data description is not sufficient. What is needed is a DDL which encompasses the physical, logical, and relational aspects of data descriptions. The DDL could be used to develop logical descriptions of which are normally stored in the data processing system, as well as descriptions of input data which are introduced into the system during execution. Thus it could be used to convert data between their internal and external form. The DDL represents an extension of the facilities provided by conventional operating systems and programming languages. By either an extension of existing compilers or by preprocessors it would allow much more explicit logical data description within source programs than is now provided.

The current tendency among data processing centers, even among those engaged in *real time* processing, is to utilize vendor-supplied operating systems. Therefore it is not sufficient to address the DDL problem at the programming language level since each manufacturer's version of the same programming language uses his own operating system facilities for producing external files.

While each organization may structure its data base and data management system to suit its own needs, having a standard DDL will facilitate the transfer of data files between organizations. It is intended that the DDL be interpreted by each such data management system and that standards be developed by which a DDL description of the file be included as part of data file interchange.

Interchange of Computer Programs

The current technology encourages and usually even forces the programmer to embed implictly in his programs many details of the equipment on which they were implemented, such as word length, character code, etc. In order to transfer a program for use in other equipment, reprogramming is generally required to reconcile the differences. For complex programs, such changes almost amount to a complete reprogramming effort. The extension of current techniques by the DDL will reduce the effort required to transfer both programs and data through making provision for the explicit logical specifications of data; this will be made possible through the uniform use of a standard DDL. Relating the logical description to the physical environment is accomplished by the data descriptions stored in the processors themselves, thus curtailing the reprogramming effort involved.

Data Independence

There is a need to be able to describe data in a standard way which is independent of the particular programming language employed. Present means of description are, for the most part, wholly contained within specific programming languages and are based on the particular computer system to be used. A DDL which is completely independent of programming languages would provide much impetus toward development of truly problem-oriented languages. Currently, data must be defined in terms of the specifications of the particular programming language being used, with the result that the same data are required to be defined in several different ways, depending upon the number of languages used to process them. A by-product arising from the development of a standard DDL will be the improvement of communications among the human elements involved in the data systems. Today there is still considerable confusion created by such terms as *format, record, data element, file, nesting,* etc., in spite of extensive standards efforts in the development of vocabularies and glossaries.

Common Use of Large Data Bases

There is a need for a DDL to describe files in both syntactic and semantic aspects in such a manner as to provide for a suitable mapping to appropriate physical structures. Such a need is predicated upon the multiplicity of users and processes which may be involved with large data bases. A formal means by which data bases can be described would facilitate communications between problem initiator and support programmer and would make possible the effective transfer of data between processes on the same or different computer systems. The semantic aspect of data is perhaps the least well defined but may well be the area which offers the promise of yielding the greatest results from investigation.

Conversion for Nonstandard to Standard Systems

The conversion of files may be as complex and expensive a task as the conversion of programs. A DDL in which a user may describe explictly his present files (which, at present, usually admit only to implicit descriptions) and the form of his target standard files will facilitate this conversion. To do this, the DDL must address itself to describing the detailed structures of the data types now in use.

In summary, a standard DDL is needed to interchange data and computer programs more effectively among independent data systems; and to make data independent of machines and associated processes. In addition, a DDL would provide better access to the data contained in large centralized data bases and would by design provide the means of converting data represented in a nonstandard form to that of the standard.

APPENDIX C

FINAL REPORT OF THE TECHNICAL SUBGROUP OF THE X3 AD HOC COMMITTEE ON DATA DESCRIPTIVE LANGUAGES

H.R. Morse
Chairman, Subgroup 3

The technical working group met twice, and various members held informal discussions at other times. Some opinions were placed on paper, and committee reports were written and discussed; all of the areas discussed were interesting, showed many paths to further fruitful discussion, and pointed out some of the problems requiring attention in the general area of the description and use of data.

These interactions could be described in many levels of detail, providing the author of this report with a rather large problem of selecting a level of detail. I have not been able to resolve that problem. Previous attempts at writing the committee report have all resulted in the beginnings of what could only be a series of volumes on the uses of data, the problems involved, solutions to various aspects of the problems, and speculations and interesting discussions of the implications of any or all of the above.

As a result, this report is fairly general in nature, only pointing out areas for fruitful work and does not delve into technical detail. I would also like to point out that the report represents my conclusions resulting from committee discussions; the same discussions may have led to other conclusions by other participants. In the area of understanding the problems involved, we must consider the following issues:

1. An understanding of *data* is required. Its use, the implications of its form on its environment, and the requirements placed on its form and structure by the environment must be better understood before any tools,

techniques, or standards can be developed which will have general application. This understanding is also required before we can provide means to ascertain, in any particular instance, what techniques will be best suited to serve the needs of an environment.

2. We must assume that there is not, nor will there be in the near future, a single technique for describing and defining data or data structures. This assumption will permit the profession to concentrate on the delineation of areas of interest or scope, within which techniques may be developed. This requires that we are able to define areas in a reasonably precise fashion, and are able to develop techniques by which we can determine what, of a set of available techniques, can be used to meet the expressed requirements of the area.

3. We must become aware that efficiency plays the rather large constraining role that it does. In this broad sense, efficiency spans the range of efficient use of communications lines when transmitting data to the efficiency of human-human communcations, over long periods. Data must satisfy many constraints in this area, including efficiency of storage (space utilization and access), efficiency of use of processing equipment, ease of transmittal to similar equipment in similar environments, transmittal to foreign environments, ease of preparing programs to process the data, ease of describing forms and useage to others, including machines and humans, and permanence of the data and data forms.

   Identification of the various parameters which a particular collection of data must satisfy, and relating this to the variety of methods of defining and implementing the data and data structures appear to be relatively untapped areas. The problem has been attacked in a local way in instances (access time versus space in disc file storage schemes is perhaps the most common) but has seldom been considered from the larger point of view which includes energy (human and machine) to transform data to other media, for use on other equipment, viability of formats for related applications, etc. One of the greatest deterents to a larger approach is the inadequacy of the evaluation methods combined with the human propensity for solving immediate problems first.

4. A stronger reliance on the formal methods and the development of formal methods related to the problem in the area must be developed. Answers to questions concerning the ambiguity of a definition or the things it defines must be available. A clear understanding of the scope of a description or the constraints under which certain properties hold must be available. Means of determining what techniques apply in specified circumstances must be developed.

   Decisions on what techniques can be used with satisfactory results must be based on an understanding of those techniques and the environment in which they will be used, and the understanding must be based on firmer grounds than intuition and experience.

5. A realization that there is not any central authority controlling or coordinating the activities in this area should encourage the development of paths of communication which will encourage the dissemination of information in a rapid, timely and precise manner among the many parties doing development work, evaluation work, and gaining experience in applying techniques. There is a great deal of work currently going on in the area and related areas, and interest seems to be increasing at a rapid rate. In this particular case, providing communication between related areas, so that work in mechanical languages can be used to advantage in the field of

data definition, for example, must be recognized as an important goal. The communication goal is doubly important, to insure the opportunity that development efforts in related areas may become aware of each other at a sufficiently early date that the possibility of influence exists.

In the area of technical activities, at least the following questions arise:

1. The problem as a whole is too large to handle – a reasonable subdivision must be made. The questions of whether one such separation into levels is suitable for all environments, whether such a structuring can be hierarchical or must have some other structure, are not resolved. It is apparent that this structuring is necessary for other reasons. In some sense, it has already begun, as evidenced by current and past activities in standardization of character sets and message formats for communications. Use of a standard (i.e., predefined) character set provides many advantages in many environments. It can be used as a basis for hardware design. We could design line printers which will accept any character sets, even permitting a dialogue which defines bit patterns, relates them to graphics and other operations such as page and line formatting. Efficiency constraints such as hardware cost and compatibility at the character set level have led us down other paths, however (see 3 above).

   The selection of a predefined set of characters is an example of separating an area and accepting a specific technique in that area. Other divisions must be made, and the techniques to be used within investigated. The limits of such divisions may depend upon the development of techniques. The way through this problem was not clear during the deliberations of the committee.

   An additional benefit of such separation is the opportunity it provides for the identification of areas which warrant different types of activity. For example, standardization activities have been taking place at the character set level for many years now. Some areas will be fruitful areas for development work – such as techniques for the definition of data and data transformation.

2. Data definition is not data transformation. Data may be defined for many purposes – for example, human communication, machine-machine interchange, and process direction. The particular use places different requirements of the definition. The use of a definition to direct or define the process of data transformation places a stronger requirement for preciseness, completeness, and a lack of ambiguity than the use of a definition to define a file to an applications programmer. Yet both are necessary and are not necessarily satisfied by one form of definition. The problems of data definition and the use of such definitions are separate but connected ones. However, providing a data definition language does not imply that data transformation is possible. We must also recognize the difference between a definition for *processing* and a definition for *recognition*. A FORTRAN program can be considered a *process* definition, BNF is an example of a *recognizer* definition. Both have their uses, require quite different facilities (processors), and provide different capabilities. Development of adequate techniques, as well as means of evaluation and selection of the appropriate technique to use in any specified circumstances, must be developed.

3. Means of making best use of current techniques, and of providing transition means to better techniques, as they are developed, are necessary. Investigations should start with existing facilities and remain aware of them.

4. This area is a fruitful area for standardization activities at some levels, but standardization in the classical sense is not the simple full answer. The development of a collection of tools and techniques, means of evaluating them, means of selecting the appropriate ones for an environment, means of describing the environment all must be developed. Included in this collection will be standards (as there are already – USASCII for example).

The technical areas require considerably more work before any meaningful approach can be taken. This may be accomplished in many ways. A standardization activity to monitor developments and initiate activity whenever standardization within an area becomes potentially fruitful is desirable. The problem of separating the problem into areas more amenable to attack must be addressed by a collection of people conversant in the area. The services of the newly formed ACM SIC File Description and Transformation (SICFIDET) have been offered to perform developmental work, and this is one area in which it might fruitfully pursue activities. Well defined developmental activities, which are not being done other places, should be initiated, but I cannot suggest how within the standards framework. The problems of communication should be pursued by the establishment of journals, newsletters, symposia, and conferences – all aimed at creating an atmosphere in which exchange of ideas and information can take place readily, rapidly, and without prejudice. This is also an area in which professional bodies, such as the new ACM SICFIET, will be active. Ways of encouraging and supporting those activities should be pursued. Enough of a standards framework should be established to monitor activities when necessary and to provide guidance to the computing community in areas related to potential development of standards.

REFERENCES

1. Knuth, D.E. *The Art of Computer Programming, Vol. I: Fundamental Algorithms* (Addison-Wesley Publishing Company, Reading, Mass., 1968).

INFORMATION STRUCTURES: TOOLS IN PROBLEM SOLVING

M.E. D'Imperio

I. BASIC APPROACH AND FRAME OF REFERENCE

A. Computer-Oriented Problem-Solving as a Subset of General Human Problem-Solving

The basic approach, in this paper, to the topic of information structures is through the phrase *Tools in Problem Solving*. Human problem-solving activity is the crucial context, within which information structures are regarded as *tools*, actively chosen or designed by a human problem-solver in pursuit of a goal. Computer-oriented problem-solving is a restricted subset of human