should be replaced by a semicolon.

\[ \sum_{k=1}^{n} (-1)^{n-k} (\sum_{i=1}^{k} (n, k)) \]
CONTENTS

Chairman's Message ......................... 1
Editor's Remarks ............................ 2
SICFIDET Meeting Report - SJCC '69 .... 3
CODASYL Meeting Report .................. 4
MAC-SICFIDET Seminar Report .......... 5
Reports on Standards ....................... 8

Papers:

Draft: Data Description Language .......... 9
Preliminary Report on a Data Description Language .......... 14
Mooers' Comments on TC15 DDL Draft Report .......... 17
Efficiency and the Use of Data Definition Techniques .......... 20
H.R. Morse
A Taxonomy of Data Definition Languages .......... 24
T.W. Olle
Data Description Languages, Extract from: CODASYL .......... 29
10th Anniversary Meeting
Data Descriptive Languages .................. 31
Calvin N. Mooers
Report to X3 DDL - "External vs. Internal Formats" .......... 37
Calvin N. Mooers
Message Forms in Computer Networks .......... 38
D. Bobrow, R. Sutherland
Introduction to Finite State Representation of Interactive Languages, D. Hatfield .......... 43
R. See
Finite State Representation of Interactive Languages .......... 44

Sketches ...................................... 47

Publications ................................... 50

Interesting Happenings ....................... 51

Authors and Coordinates

Daniel Bobrow Bolt, Beranek & Newman, 50 Moulton St.,
Cambridge, Mass. 02139
G. Gibson International Computers, Ltd., Cavendish Rd.,
Stevenage, Hertfordshire, England.
Wm. C. McGee IBM, Palo Alto Scientific Center, 2670 Hanover St.,
Palo Alto, California 94304.
Calvin N. Mooers Rockford Research Institute 140 1/2 Mt. Auburn St.,
Cambridge, Mass. 02139.
H.R. Morse Agrippa-Ord Corporation, W. Acton, Mass. 01720
T.W. Olle RCA, Information Systems Division, Riverton, N.J.
Richard See National Library of Medicine, Rockville Tpke.
Bethesda, Maryland 20014.
W.R. Sutherland Bolt, Beranek & Newman, 50 Moulton St.,
Cambridge, Mass. 02139.
Chairman's Message

File description and translation - data description language-data
definition - program and data transferability - automatic reformatting. The emergent overlapping terminology reflects a convergence of interest in the problems of explicitly defining the features of data sets (which may include programs) and providing compatibility through a combination of data reformatting and intermediate access procedures.

At the same time there is an increased interest in clarifying the use of information structures themselves and cleaning up the interface between the procedure and the information it manipulates. It has been suggested that compatibility among separate programs and systems can be achieved easiest through a greater separation of procedures and data in future problem-oriented languages and systems.

SICFIDET will attempt to provide a means of communication for all these concerns as well as information for those who want to learn more about the data and storage structures used today. The bulletin will try to bias itself in the direction of the timely rather than the polished, and will reference longer articles and books by abstracts. Especially the Committee will try through the bulletin to describe work planned or in progress, and particular problems which we feel should be of general interest.

As description techniques and mapping techniques become formalized, SICFIDET will encourage the publication of papers and will sponsor or co-sponsor working sessions and tutorials to promote thier understanding and their use.

But within this general framework the direction of the Committee depends on you and the uses you will make of it. There are many tasks that can be taken on or sponsored as a Committee. There are the allied problems of compiling a glossary of data terminology and of data structures. There is the question of the universality of description schema and the compatibility of different description systems, and its implications for standards, which brings up SICFIDET's relation to standards setting bodies such as USASI and ECMA. There are many practical problems which need better definition, such as the automatic construction of descriptors by programs which generate the data, or the form of a translation defining language, or a possible standard for description of the various mathematical forms of matrices.

So since the future of SICFIDET is up to you I would, on behalf of the Committee, like to hear from you on what you would like to see us do beyond the stimulation of communication exchange.

Don Hatfield - Chairman, SICFIDET
EDITOR'S REMARKS

As editor of this first issue of fdt, I can think of no better time to state our idea of publication policy, and of the role we would like this publication to play.

The area of data and file definition is concerned with the capability of defining data and data structures. With a broad definition of the work 'define', we include a wide range of techniques, from the written explanation of the potential use of a data base to a user, to the theoretical problems of determining the completeness and degree of ambiguity in a definition of a data structure which (definition) will be processed to produce tables or code to be used to build or access that data structure.

The diversity of people interested and working in the areas is wide indeed. The terms they use, the meanings these terms connote, and the context in which the people discuss the problems present a ready made environment for communications breakdown. This couples with the fact that many people do not naturally encounter people from different environments who are working on similar problems, albeit using different terminology. We have, I believe, a ready made situation for encouraging redundant and conflicting efforts, in a field with sufficient redundant and conflicting efforts to satisfy its minimum yearly requirement for some time to come.

The first goal of this publication is to serve as a communication medium, especially between people of diverse backgrounds and interests. We will attempt to indicate where work is in progress, and what kind. We intend to give short sketches of people working in the area and their interests. We will report on relevant developments in the standards area, here and abroad. We will publish announcements of relevant meetings. We will publish informal communications, with as little editorial filtering as possible in the hope of encouraging the rapid dissemination of information. We will attempt to review anything of interest, and provide pointers to other reviews.

We will solicit and accept articles for publication which are less formal than required by more polished publications, in the hope of bringing embryo ideas and approaches to the public for comment, criticism, inspiration and information on a timely basis.

Finally, and significantly, the publication will be shaped by the expressed needs of the audience. So keep those letters and cards coming folks; we need your support and direction.

H. R. Morse
First SICFIDET Meeting, SJCC '69

A combined panel and business session was held in Boston this spring. With four talks an attempt was made to delineate the major difficulties in developing useful data descriptions. The four areas covered were the stratification of data definition (T. W. Olle, R.C.A.), data description and data independence (W. C. McGee, I. B. M.), efficiency considerations in data description and translation (H. R. Morse, Agripa-Ord), and the relations between data and procedure (Jorge Rodriguez, E. A. E. project, M. I. T.). Two of the talks are printed in this issue of the bulletin and two will be included in the next issue.

At the business meeting which followed a topic was discussed which has also come up at the USASI Ad hoc Subcommittee on Data Descriptive Languages, at CODASYL, and at the first meeting of the Data Description Seminars at project MAC. This is the problem of making a survey of data bases, structures and terminology when there is no default language for the survey. The lack of any accepted language for talking about data at this point seems a major block to a better understanding of the data structures in use today.

Second SICFIDET Meeting, ACM '69

The SICFIDET Panel will present two complementary aspects of data description (or definition). On the one hand the needs of large information retrieval, data management and data analysis systems will be examined as they relate to explicit description and transformation needs for data files. Some specific data incompatibilities which already exist will be described in detail.

The second area of discussion will be an aspect of the requirements for Data Descriptive Languages—the difference between logical and physical forms of data and the problem of how this difference can best be stated.

Consequent of this difference between content and implementation is the implication that a DDL must be context sensitive. An attempt will be made to survey and delineate the context sensitivities arising from a multi-level (in the sense of T. W. Olle) definition of a data-file, and suggest how rules for these sensitivities can be incorporated into a DDL with some generality.

The session will be chaired by Mr. W. C. McGee of the I. B. M. Palo Alto Scientific Center. It will be held Wednesday, August 27 at 5:00 p.m. at the San Francisco Hilton.
A Report on the CODASYL Meeting from A. K. Bhushan

Project MAC, M.I.T.

TO: Dr. J. C. R. Licklider

FROM: A. K. Bhushan

SUBJ: CODASYL Tenth Anniversary Meeting

On May 27-28 I attended the CODASYL (Conference on Data System Languages) Tenth Anniversary Meeting held at the Statler Hilton Hotel, Washington, D.C. The aim of the two-day meeting was to review briefly the achievements of CODASYL over the past decade and primarily establish a direction for the coming decade. The sessions included a discussion of software standardization, future software developments, procedural and non-procedural languages, data descriptive languages, data management languages and systems and the future roles of COBOL and CODASYL. A questionnaire was circulated to determine the attendees' responses on these subjects.

The CODASYL systems committee has just completed a survey of generalized data base management systems. Copies of this 400 page survey were distributed to the attendees and the report is available through ACM. This narrative survey of the typical features of 9 data base management systems is a feature-by-feature analysis in a common format and terminology.

Some time was spent on discussing COBOL and where it should be going. This discussion turned to data descriptive and data management languages and the role CODASYL should play in this development. It was recognized that USASI X3 ad hoc committee and the ACM Special Interest Group on file description and translation are also working on the problem.

It is my opinion that CODASYL is too preoccupied with COBOL and its concept to do justice to an effort towards a standard data descriptive language. However, it can constitute a valuable input to the USASI X3 group in its work and ACM SIGFIDET can provide some of the analytical background. In addition, the narrow reach of the CODASYL group makes it unsuitable for this task that involves the computer community at large.

Editor's Note: The CODASYL survey may be obtained for $7.00 by writing to ACM, SICFIDET, c/o ACM, 1133 Avenue of the Americas, New York, New York 10036.
Announcement of the Project MAC

Data Description Seminars

Dear Colleague:

This letter concerns a proposed seminar and a proposed book in the area of data description and translation. The plan involves a weekly seminar, the collection and publication in inexpensive book form of a set of papers, and then a second weekly seminar during the academic year.

The summer seminar series will be held under the ageis of Project MAC and the Special Interest Committee on File Description and Translation (SICFIDET) of the Association for Computing Machinery. Each week, one or more people will present pertinent material, which will then be the subject of group discussion. The purposes of the summer seminar will be:

1. To disseminate informally the current ideas and results in the field of data description and translation.

2. To stimulate thinking and encourage further work.

3. To define and organize a seminar series for the academic year 1969-70--in which problems of data description and translation would be studied in depth.

4. To establish a mailing list and a communication system for dissemination of information in the field of the Special Interest Committee.

The material presented and discussed in the summer seminar, according to the present plan, is to be turned into a book. The idea is to put the book together quickly and informally and to publish it inexpensively. The purpose is to collect together in one place a large fraction of the ideas and results that are extant so that they will be readily available as source materials for forthcoming efforts to make some real headway on problems of data description and translation.

According to present plan, the seminar series to be conducted during the academic year will not be a formal credit seminar but will be a working seminar open to participation by everyone with a serious interest in research on data description and translation.

Seminars will be scheduled at 2:30 p.m. on Mondays throughout the summer. If you are interested in participating write or call Miss D. Scanlon, 617-864-6900 X 5851.
Summaries of the first and second Data Description Seminars (since it is planned that transcripts of the seminars will be later compiled into book form, these summaries are purposefully brief. Further information may be obtained by contacting project MAC)

First Seminar

---J. B. Dennis
Project MAC

Prof. Dennis drew a parallel between data transferability and program transferability. In the case of the program to be transferred from environment 1 to environment 2 there exists an abstract program (say in a source language) which can be translated into a form that can operate in either environment. Similarly, there must be an abstract representation of the data (and this means machine independent) in order to move data between environments. The problem of designing a reference version of the data, or reference language for representing data structures, is just as difficult a problem as designing a universal programming language, and if you want the data structure to be operable on my arbitrary programming languages, is equivalent.

A description of the data structure implemented on MULTICS (the project MAC time sharing system) followed. It was described as a tree with multiple paths allowed to a node but not circuits. The naming rule for nodes precludes two nodes with the same name leaving a branch. Prof. Dennis proposed such a structure as the basis for a generalized data structure. He is developing a set of production rules for defining and handling such trees. A discussion of the relation of such structures to common non-tree forms of data (matrices, graphs, text strings) concluded the presentation.

First Seminar

---Dr. D. Yntema
Lincoln Labs

Dr. Yntema described the coherent programming system at Lincoln Labs. This system is one in which there is file sharing, and file validating through checking description codes found at the beginning of each file. In its design four rules were followed with respect to data handling: 1. allow for unforeseen data structures to be put on the system; 2. allow for such expansion without replacement of existing software; 3. make error conditions intelligible to the user; 4. allow the program to blow on a malformed file.
The files themselves are categorized into about 10 types, the principal distinction being that between text and multidimensional arrays of rational numbers, plus a special type for "others". The job of translation resides with the individual applications programs as do any consistency checks. Human pressure keeps descriptions consistent.

Second Seminar
---Joseph Weisenbaum
Project MAC

Prof. Weisenbaum delineated the differences between facts, data, data structures and storage structures. He observed that much of the confusion over data description is due to the failure to make the distinction.

Facts are the primitives but useless (and only really defined) in light of a hypothesis, when they become data. For a different hypothesis a collections of facts can yield a different set of data. The structure of the data again reflects a hypothesis as to how the data is interconnected for some purpose, and is itself data. Then there is the storage structure which has to do with how the data structure is modelled on a computer. The storage structure is a function of the modelling of the data structure, and a function of the hardware and software facilities the programmer has available. This can result even in non-sequential physical core addressing. It's important to remember that the programmer has some freedom in choosing a storage structure for the data structure. And an important question is, can one find automatic means for getting from one to the other.
REPORT ON STANDARDS

Introduction

This section contains reports on standardization activities related to data and file definition. The activities of initial and obvious interest are the USASI/X3-Ad Hoc Committee on Data Descriptive Languages (X3 DDL), USASI, X3.8 - Data Forms and Formats and ECMA/TC15.

We will report on meetings, review or publish interesting materials distributed by the committees, and attempt to print information about future activities. In this first issue of fdt we will state the scope of X3-DDL, and print two draft reports from ECMA/TC15 and a comment on one by Calvin Mooers.

Definitions

For the next issue, we will prepare a collection of definitions of committees and organizations in the standards area for background information. Look here to find out what the following mean:

ECMA, X3, TC15, USASI, BEMA, X3.8, Z39, ISO, X3.4, CODASYL, TC97/SC2, IAG, JAG.

The first two people supplying a complete list of correct definitions will be given a free two years subscription to fdt.

X3 Ad Hoc On Data Descriptive Languages

The committee was formed earlier this year under the chairmanship of John Gosden, to investigate and report to X3 what, if any, standardization effort should take place in the area of Data Descriptive Languages.

The main committee was composed of members of standards committees working in related areas and other computer professionals with an interest in the area. At the initial meeting, three subcommittees were selected to do detailed investigation in three specific areas:

0) Need for standardization
1) Current work in the area
2) Technical considerations

A final report, written by Chairman Gosden from the reports of the three subcommittees, has been submitted to X3. If X3 permits, the report will be published here in the next issue.

ECMA/TC15

In this issue, we have two draft reports. The first is a very informal communication by G. Gibson to the committee. The second is a draft report of the committee. Please note that these are both working papers, published here for information only. Please note also that the Gibson draft discusses a data description language and not a data manipulation language. The third document was prepared by Calvin Mooers in response to the second report.
1. General

This draft is certainly incomplete; I have not stated the way in which "macro entries" are inserted in a "description" and there is no provision for "if clauses". There are probably other errors as well. Further facilities which must be added are:- comment clauses, ability to deal with superimposed hierarchies, de-subscripting.

2. Syntax

The syntax is given on the attached sheets.

3. Semantics

The semantics is given only when it is not, hopefully, obvious.

3.1 Numbers

This is the appropriate sub-set of the Algol syntax. % INTEGER% obtains the value of the data field to which reference is made and uses this as an "unsigned integer"

3.2 Identifiers

Identifiers are made up by a sequence of one or more characters of which the first is a letter and the last, if more than one, is a letter or a digit.

3.3 Names

A "general name" refers to an entry in the Data Description. A "name" refers to an item in the data which may consist of one or more fields. A "subscript" identifies the particular occurrence of an item which is repeated as specified by a "rep clause". Only sufficient identifiers need be given to indicate a unique path through the hierarchy, and must be given in order going from the most general to the most particular.

3.4 Pictures

Picture characters have the same meanings as in Cobol descriptions; other ones should be added to the list. % PICINF % uses the contents of a data field as a sequence of picture characters.

3.5 Quotes

% LITERAL % uses the contents of a data field as a quote string.
3.6 **Strings**

A string expression is the concatenation of the string alternatives.

3.7 **Terminations**

The termination of a data field is given by the occurrence of the last of a sequence of marks. These marks are either the occurrence of a given number of characters of any kind or the occurrence of a particular sequence of characters specified by a string secondary. In the latter case these characters may be either included in or excluded from the data field.

3.8 **Normal Clauses**

A "size clause" gives the maximum length of a data item and is required for the allocation of storage.

3.9 **Shorthand Clauses**

% LENGTH %, %QUOTE %, and % PIC % enable certain simple and frequently occurring types of description to be expressed in a shorter form. % PIC % can only be used when any "number expressions" in the "picture string" are "unsigned integers". The "unsigned integer" in the expansion is the number of characters in the string.

3.10 **Entries**

An elementary entry is an entry describing an item which is not sub-divided further in the given description. Such an entry must give information to define the end of the item and the maximum length of the item.

3.11 **Description**

The level number of an "entry" is given by the "unsigned integer" with which the "entry" commences. If a "hierarchy" consists of "entry" followed by "mult" the level number of the "hierarchy" is given by the level number of the "entry" which must be less than the level number of the "mult". The level number of a "mult" is given by the level numbers of all hierarchies in it which must be the same.

3.12 The syntax as given does not indicate how to identify the occurrence of formal parameters in the macro detail.
3.13 Substitutions

The "ref sub" calls for the insertion of the string of characters between % MACRO % <MACRO DEPN> and the corresponding % END %, the existing level number being added to level numbers in the string, and formal parameters being replaced by actual parameters. The "like sub" calls for the insertion of the string of characters comprising the hierarchy given by the general name except for the entry head, level numbers being altered by the difference between the existing level number and the level number of that hierarchy. The "given by sub" calls for the insertion of the string of characters contained in that data item, the existing level number being added to level numbers in the string.

G. A. Gibson
19.5.69
1 NUMBERS
1.1 <UNSIGNED INTEGER>:<DIGIT><UNSIGNED INTEGER><DIGIT>
1.2 <NUMBER>:=<UNSIGNED INTEGER><NUMBER><NAME EXP>!<NUMBER EXP>
1.3 <ADD OP>=:+
1.4 <MULT OP>=:*/
1.5 <FACTOR>:=<NUMBER><FACTOR><NUMBER>
1.6 <TERM>:=<FACTOR><TERM><MULT OP><FACTOR>
1.7 <NUMBER EXP>:=<TERM><ADD OP><TERM><NUMBER EXP><ADD OP><TERM>

2 IDENTIFIERS
2.1 <IDENT CHAR>:=<LETTER><DIGIT><UNDERLINE>
2.2 <IDENT TAIL CHAR>:=<LETTER><DIGIT>
2.3 <IDENT HEAD CHAR>:=<LETTER>
2.4 <IDENT TAIL>:=<IDENT TAIL CHAR><IDENT CHAR><IDENT TAIL>
2.5 <IDENT>:=<IDENT HEAD CHAR><IDENT TAIL>

3 NAMES
3.1 <GENERAL NAME>:=<IDENT><GENERAL NAME><IDENT>
3.2 <GENERAL NAME EXP>:=<GENERAL NAME>
3.3 <SUBSCRIPT>:=<NUMBER EXP>
3.4 <NAME EL>:=<IDENT><IDENT><NAME EXP><NAME EL>
3.5 <NAME>:=<NAME EL><NAME>,<NAME EL>
3.6 <NAME EXP>:=<NAME>

4 PICTURES
4.1 <PIC CHAR>:=9A%I
4.2 <REP PIC CHAR>:=<PIC CHAR><PIC CHAR><NUMBER EXP>
4.3 <PIC STRING>:=<REP PIC CHAR><PIC STRING><REP PIC CHAR>
4.4 <ACTUAL PIC>:=<PIC STRING>
4.5 <PIC PRIMARY>:=<ACTUAL PIC>%PICINF%<NAME EXP>

5 QUOTES
5.1 <QUOTE CHAR>:=ANY MEMBER OF COMPLETE SET OF CHARACTERS
5.2 <QUOTE STRING>:=<QUOTE CHAR><QUOTE STRING><QUOTE CHAR>
5.3 <ACTUAL QUOTE>:="<QUOTE STRING>"
5.4 <QUOTE PRIMARY>:=<ACTUAL QUOTE>%LITERAL%<NAME EXP>

6 STRINGS
6.1 <STRING PRIMARY>:=<PIC PRIMARY><QUOTE PRIMARY><STRING EXP>
6.2 <STRING SECONDARY>:=<STRING PRIMARY><NOT><STRING PRIMARY>
6.3 <STRING ALT>:=<STRING SECONDARY><STRING ALT><OR><STRING SECONDARY>
6.4 <STRING EXP>:=<STRING ALT><STRING EXP>,<STRING ALT>

7 TERMINATIONS
7.1 <BOUNDARY>=%INCL%<EXCL%
7.2 <STRING BOUNDARY>:=<BOUNDARY><STRING ALT>
7.3 <MARK PRIMARY>:=<NUMBER EXP><STRING CONDITION><TERM EXP>
7.4 <MARK>:=<MARK PRIMARY><MARK>%OR%<MARK PRIMARY>
7.5 <TERM EXP>:=<MARK><TERM EXP>,<MARK>

8 NORMAL CLAUSES
8.1 <REP CLAUSE>:=%REP%<NUMBER EXP>
8.2 <STRING CLAUSE>:=%STRING%<STRING EXP>
8.3 <TERM CLAUSE>:=%TERM%<TERM EXP>
8.4 <DEFN CLAUSE>:=<STRING CLAUSE><TERM CLAUSE>
8.5 <SIZE CLAUSE>:=<MAX%<UNSIGNED INTEGER>
9  SHORTHAND CLAUSES
9,1 %LENGTH%<UNSIGNED INTEGER>%TERM%<UNSIGNED INTEGER>%MAX%<UNSIGNED INTEGER>
9,2 %LENGTH%<NAME>%TERM%<INTEGER>%NAME>
9,3 %QUOTE%ACTUAL QUOTE>%STRING%ACTUAL QUOTE>%MAX%<UNSIGNED INTEGER>
9,4 %PIC%<ACTUAL PIC>%STRING%<ACTUAL PIC>%MAX%<UNSIGNED INTEGER>

10  ENTRIES
10,1 <LEVEL>:= <UNSIGNED INTEGER>
10,2 <ENTRY TIP>:= <LEVEL>@<ENTRY TIP><ENTRY TIP><ENTRY TIP>
10,3 <ENTRY HEAD>:= <ENTRY TIP><ENTRY TIP><ENTRY TIP>
10,4 <ENTRY DETAIL>:= <ENTRY TIP><ENTRY TIP><ENTRY TIP>
10,5 <ENTRY ENTRY>:= <ENTRY TIP><ENTRY TIP><ENTRY TIP>
10,6 <ENTRY>:= <ENTRY TIP><ENTRY TIP><ENTRY TIP>

11  DESCRIPTION
11,1 <HIERARCHY>:= <ENTRY ENTRY><ENTRY ENTRY>
11,2 <ENTRY ENTRY>:= <ENTRY ENTRY><ENTRY ENTRY>
11,3 <ENTRY ENTRY>:= <ENTRY ENTRY><ENTRY ENTRY>

12  MACROS
12,1 <FORMAL PARA LIST>:= <IDENT><FORMAL PARA LIST><IDENT>
12,2 <MACRO DEFN>:= <IDENT><IDENT><FORMAL PARA LIST>
12,3 <MACRO DETAIL>:= <CHAR><MACRO DETAIL><CHAR>
12,4 <MACRO ENTRY>:= <MACRO DETAIL><MACRO DETAIL>

13  SUBSTITUTIONS
13,1 <PARA>:= <NUMBER EXP><NAME EXP><NAME EXP><STRING ALT><MARK>
13,2 <ACTUAL PARA LIST>:= <PARA><ACTUAL PARA LIST><PARA>
13,3 <MACRO CALL>:= <IDENT><IDENT><ACTUAL PARA LIST>
13,4 <REF SUB>:= <REF SUB><MACRO CALL>
13,5 <LIKE SUB>:= <LIKE SUB><GENERAL NAME>
13,6 <GIVEN BY SUB>:= <GIVEN BY SUB>
13,7 <SUB>:= <REF SUB><LIKE SUB><GIVEN BY SUB>
The following represents the consensus of opinions among the members of ECMA/TC 15 with regard to the Data Description Language, abbreviated DDL.

The development of a DDL is considered to be in the interest of the members of ECMA. Guidelines for this development are summarized below.

1. Objectives

The purpose of the DDL is to provide a concise method of specifying the format of a collection of data. This will facilitate the handling of data in the following areas (listed in order of importance):

- interchange of data, viz. data transmission, transfer of recorded data from one party to another, sharing access to data with other parties;
- conversion of data from one medium to another;
- extraction of specific data from a larger collection of data.

The scope of the DDL extends from the high level of "data base", or "set of interrelated data", down to the level of "character", or "bit". Recognizable intermediate levels include "record" and "field". The DDL must accommodate data having structures like trees, arrays, lists and combinations of these, data items whose format depends on other data items, and both fixed and free formats, the latter with explicit or implicit delimiters.

The appearance of the DDL must be such that it is

- understandable for a human, so that a written format specification can be interchanged together with the data;
- readable by a data processing system, so that a format spec can be stored or translated into a machine-oriented data description.
- compact, or being able to be condensed, so that a format spec can be contained in labels that accompany the data.
The structure of the DDL must be such that it allows definition of sub-sets of the language.

2. Restrictions

It is recognized that all objectives cannot be fulfilled at once. Therefore, the following restrictions are suggested for the first stage of the development.

- the DDL shall be confined to data consisting of strings of characters chosen from the ECMA-6 alphabet;

- prime attention shall be given to the ability of the DDL to specify the format of a "record" in terms of fixed "fields", assuming that the way of isolating records already exists.

These restrictions are intended to be relaxed after a first result has been obtained.

3. Applicable documents

Available documentation which may serve as a basis for the development includes:

- Language specifications of COBOL (Data Division), PL/I (Declare Statement), Algol 68 (Mode Declaration);

- Various documents having originated from ECMA/TC 15 members and from USASI X3 sub-committees;

- ISO Draft Recommendations on Terminology prepared by TC97/SC1.

4. Method of development

The existing methods of describing a character string provide some of the information required for the various objectives given above but none of them supply all the information necessary in all cases. The first tasks are to define a restricted objective for a DDL, to determine the properties required to meet that objective, and to develop from existing methods of data description a DDL with these properties. A suitable restricted objective is the ability to describe the way in which a record is composed of fields. This description must be such that it is possible to transform it into the description required for a program written in any of the common higher-level languages. This will enable a file which has been produced by one system to be processed by a different system.

It seems probable that the initial attempt at a DDL will be able to deal only with a limited range of structures. Until
the DDL is developed it is not possible to state the other restrictions which may prove necessary but a probable one, for example, is that the fields of a record must not overlap. The next task therefore must be to develop the language to remove those restrictions which prevent practical use of the DDL. Further development can then extend its use to more general structures and to more general applications.
USA STANDARDS COMMITTEE CORRESPONDENCE  
June 5, 1969  
X3 Comm. Data Definition Language

Mr. John A. Gosden  
The MITRE Corporation  
1820 Dolley Madison Blvd.  
McLean, Virginia 22101

Dear John:

Thank you for forwarding the copy of Mr. Gibson's draft prepared for TC15 for a Data Description Language. It appears to be a very excellent foundation, and I especially agree with his choice of a macro base for a DDL. I also concur with Mr. Gibson in that it would be most desirable to have a high order of trans-Atlantic cooperation so that a common result may be secured.

The most significant paragraph to me in Gibson's draft was the first where he mentioned some of the topics which he had not dealt with. I felt that he had wisely truncated his draft at the point he did because of the significant problems that are next in line. It is in this area that I wish to make some comments.

A "data definition" is a part of a transformation which mediates between useful "output data" and supplied "input data". Often data will be selected from the input data, with the remaining data being ignored. Thus the transformation between the output data and the input data may be singular or irreversible. The entire transformation will be defined by a "processor" (machine plus software), the data definition, and usually some "control parameters" to define the desired output selection and format. The data definition language cannot be considered in isolation. In many cases, the data definition will be in two or more parts, a "local part" defining how the user wants the output data to be put into a format, and a "data part" which is part of the input data. To further complicate matters, the "control parameters" may in fact be segments of actual data definition language, and not merely numbers or literal strings. In an interactive on-line environment, the control parameters may be supplied at run time. Taking this expanded viewpoint, the following are some of the extended aspects that I believe need attention in the completion of the sketch supplied by Gibson. (My points are ordered in pedagogical sequence, not necessarily in logical sequence.)

1. There is the need to have an option to expand, or not to expand, a macro call within another macro detail.
2. If the expansion of a macro contains a macro call, a similar option must prevail.
3. Presumably a macro call can be within a parameter, and again the time of its expansion must be under control.
4. The parameters of a macro call may in fact be the entire record of the input data, or some portion of it. A symbolic mechanism for handling this is needed.

5. Macro definitions may occur at any time, i.e., they may be made in the local part of the data definition, may be in a local store or library available for call and expansion, or they may be made as a result of some portion of the data part of the data definition.

6. In a strict analogy to macro calls, there exists the same need for options in regard to the time of expansion of Gibson's "number expressions" and "name expressions". Further, since these may occur within a "macro detail", both of these may contain formal parameters of the containing macro. They may also be expressed in terms of a macro call.

7. Since the input data cannot be guaranteed to be free from error, there is need for provision in the data definition to indicate when and how to take action (branch or output a diagnostic) when an error is encountered.

8. Since the occurrence of a diagnostic may require supervisory intervention, the data definition statement should have provision for auxiliary input and re-start for such a case.

9. It can be expected that portions or segments of the data elements of the input data will be required. These would be short segments of a complete literal string, or a few digits from a longer unsigned number, etc. The data definition language should be able to define such sub-elements within the input data elements. Such sub-elements may be required as input parameters for the required data transformation, or for the desired output.

10. Conversely, it may be necessary to concatenate sub-elements to produce an actual parameter or to produce output data. The data definition language must permit description of such concatenation.

11. The general problems of "de-subscripting" and "superimposed hierarchies" may find their solution from taking a more general viewpoint of "entries" in accordance with my comments under points numbered 9 and 10 (and others) in my list.

12. Most of the preceding points have profound implications in regard to the nature of the process, or processor, which is implied by the data definition language. These implications must not be overlooked.

A general and complete data definition language must, I believe, cope with the range of problems including the above points. While a goal of this generality would appear to be most formidable and speculative, it is not actually so. Much of the work toward a DDL of this generality is already provided in working example by my macro-based TRAC language. Cf. Mooers and Deutsch, Proc. 20th Natl. Conf. ACM pp. 229-245 (1965), and Mooers, CACM v.9 pp. 215 - 219 (March 1966). It would not be difficult to carry over
the logical techniques of the one into the form of the other, e.g. over into Gibson's format.

However, what will be more significant to the typical user will be various truncations and derivatives of a complete language. It is my belief that it is wisest first to develop a clean and logically complete language, and then to truncate it and to derive from it for various uses. The alternative, that of trying to patch up and to extend an incomplete formulation, is generally a method for seeking trouble. The extensions tend to lead to unpredictable consequences and interactions.

To reiterate, I believe I like the way in which Gibson has begun to lay down a logical foundation.

Sincerely,

Calvin N. Mooers
Efficiency and the Use of Data Definition Techniques
by: H. R. Morse

I thought I'd start by stating four points and then discussing them to some extent.

1. There are many definition techniques currently available and in use, and there are many more coming, especially with the current emphasis on explicit data definition.

2. It is optimistic to expect a single acceptable technique for data definition to be developed in the near future, or perhaps even within 10 years. I suspect that even if one were it wouldn't satisfy the variety of needs which data definition must satisfy, even today.

3. The choice of a technique or a collection of techniques to solve a particular problem is primarily a matter of cost versus efficiency. I take a fairly broad definition of efficiency, which I will discuss shortly.

4. Measuring, evaluation and selection tools have to be developed along with tools which will permit us to explicitly define the environment within which data is to be used and the criteria that the environment will place on the data before we are to usefully apply any data definition techniques.

There are many methods of data definition in use currently. One common example is writing programs. A program which processes data contains an implicit definition of that data. This technique is used far too often today. An equally common technique is to provide an English language definition which is processed by programmers to result in a machine language or higher level language definition of data. This technique does not require a rigorous definition, since a great deal of interpretation is normally done by the programmers. A more precise definition is required in a description of existing collections of data, since the freedom to interpret is not available. In some cases, such a definition can be transmitted in a form such as the data division of a COBOL program. All of the preceding examples are of "processor" definitions, i.e., a definition which describes how the data is to be processed. The alternative is a "recognizer" definition which would indicate how the data would be extracted, how items could be recognized, or how one might recognize data items on a transmission line. I have a feeling that data definitions in LISP, its friends and relatives fall into a separate category, which I cannot identify at present. The definition of data as it occurs in LISP is very strongly connected with the structure of that data. In fact, the data consists almost entirely of the structure of that data plus some of the

Presented at SICFIDET Meeting, SJCC '69
processing programs, but it is more closely connected with the structure.

Bachus-Naur Form is an existing technique useful for data definition. Although it has been primarily applied to the definition of languages, it is useful to examine some of the ways it has been used. A definition in Bachus-Naur Form has been used as direct input to tables within a compiler which drive that compiler. That is, the definition is used directly. Although I can't identify a case, I am sure a BNF definition has been used to drive a processor which generates the code that becomes a compiler to compile the language described by the BNF definition. BNF has been used as a written definition to a programmer from which to generate code which is a compiler. In that case it is a data definition which is processed by one of the less precise (predictable) but more intelligent processors we have around today - the programmer. Using such a BNF definition as a guide eliminates some of the problems of automated processing of such definitions that have not been solved.

The eventual goal is to develop and build definitions which can be included in data files as part of the file, and passed with the data from environment to environment. Such definitions can be used in a variety of ways, such as those indicated in the discussion of Backus-Naur Form.

1. The definition in the file can be effectively interpreted. It can be used in place without any transformation. A good example would be if one wanted to make a single inquiry of a data collection. Interpreting a data definition on the way into the file is very inefficient usually, and it will probably work. The definition can be used to transform the file into a canonical form required by a specific environment. This is called for when the file will have heavy use in the new environment.

2. The definition can be processed to produce code which will process that file either in its original or transformed form as required by a new environment.

3. And, as always, a definition can be used as a guide to a programmer building code to process that file. An important point is that it is not going to be possible to use a definition of a file file contained within a file to transform a file unless we have better ways of specifying the criteria for use, for example whether in a random access file it has to be indexed on a particular key, because access time via that key is critical. I see no evidence that we have thought very much about specifying those parameters as input to the process of transforming files.

I stated earlier that I would talk about efficiency. We are all used to talking about the use of storage space versus access time and optimizing one against the other for specific needs. We have even played with optimizing for ease of use, so that we
don't have to go to a great deal of effort to build and use complex data structures, usually at a cost of storage space and access time for the convenience. The cost (and payoff) can be quite great in languages like LISP. We know one can write a compiler in LISP in one typewritten page and take half the day doing it. We know it will take forever to run, but will provide an adequate test of techniques.

We also have a very gross qualitative feel for implementation effort. We know that it is usually harder to implement straightforward things in machine language than in FORTRAN or COBOL. We also know that we will pay in efficiency if we violate the basic precepts of those languages in attempting to use them for something for which they are not well suited. I don't believe we have a good quantitative measure of the cost in efficiency or gain of ease of implementation for a specific environment, nor in general.

We have some idea of the problems of maintenance. In the last four years everybody has been reprogramming like mad, so we have a lot of experience in the cost of maintaining programs across environments.

The problems of maintaining programs and the problems of maintaining data bases are, however, separate problems. We don't have a quantitative measure of the cost or effectiveness of techniques especially when related to different environments or how they meet criteria for use. We don't have much of a feel at all for the problems associated with evaluating cost versus effectiveness in preparing files for transformation. Again we are discussing problems associated with the transformation of files for other uses in other environments. Examples in this area include preparing magnetic tapes to be shipped out to a number of different installations, containing data which is well-defined and possibly even well described, to be used for different purposes in different environments. A specific example is The Library of Congress MARC II tapes. The purpose of those tapes is for the distribution of bibliographic and descriptive information on publications. I don't think that at the current time there is an effort to relate the cost of preparing the tapes, which is supported in one place, and the cost of processing those tapes later at different places.

We do know some of the quantitative measures of cost associated with transmission of information. We know how much it costs to mail a magnetic tape in a big box with stamps on it, and we know how much it costs to ship characters on communication lines. We don't know what the cost is of preparing the data for shipment at one end and undoing whatever was done to it so we can use it at the other end.

The thing of which we have perhaps the least quantitative
concept, that I cannot even identify as a quantitative concept, is the relationship between different ways in defining data and the understanding of that data. In this case I refer to the understanding of someone who is the end user of a data file or data management system. His problem is that at some point he will have to understand the implications of the variety of ways that his data may relate and what the structures mean. In general the end user of a large data base is not conversant enough with the techniques that are buried in the innards of the system to manipulate pieces of information in hard storage. I don't know how we provide the information that permits the end user to understand the implications of things he may do so he can evaluate the results of future actions, or determine a "good" way to use that data.

Finally, in the transmission or transformation of files for communication purposes there is a whole set of hard criteria or hard values that we can measure and evaluate. These include things like CPU time, the cost of the storage media, cost of transforming character sets, and the cost and efficiency of using one character set versus another. In fact, we can measure the efficiency of defining a character set versus use of one of the pre-defined sets. Less measurable, but also to be considered is the cost of using a data system or a collection of data under different languages, the cost of transforming data collections for use under different support systems, operating systems and filing systems and the efficiency that may be gained by transforming the file for a completely different use versus using it as is.

Data definition techniques are developing - they are both good and necessary. But, before we are going to be able to use them well, we have to know the implications of the various ways of definition and definition techniques. The constraints that are implied by the various ways of defining data, the environments in which they will be used, and the criteria that have to be met by their use, all require deeper understanding first, and, eventually, precise means of evaluation. The goal is that some day in the future we will have some algorithmic means for the selection of appropriate definition and transformation techniques and precise ways of specifying the criteria for use and the environments in which data will be used.
4. Session Outline

Date: Wednesday 28 May 1969. 9:30 A.M. - 10:30 A.M.

Title: Data Description Languages

Chairman: T. W. Olle

Speakers: W. C. McGee, J.A. Gosden

Outline: Most data management systems and higher order languages make use of explicit data descriptions. These descriptions permit the system to do many things automatically that would otherwise have to be done by the user, including space allocation, index building, input/output editing and conversion. Data descriptions also afford the user a certain degree of data independence, i.e., independence from the physical representation of the data, so that changes to the latter may be made without affecting the user's programs.

More recently, data description has been proposed as a technique for simplifying data translation, i.e., the conversion of data from one logical/physical form into another. If techniques can be found for appropriately describing the source and target forms of the data, the process of translating from one form to another is meaningful and useful.

Data description may be made on at least four levels:

a. A conceptual description (e.g., employing graph-like diagrams) which facilitates human communication but is not necessarily intended for machine processing.

b. A logical description which conveys to the system the logical properties of the data, e.g., item names and types, item grouping, and group relationships.

c. A physical description which specifies the manner in which the data is stored in a given machine configuration.

d. A coded description which is a machine language distillation of the user's logical and physical descriptions, plus additional information required to process the body of data being described.

Most work to date has been carried out at the logical description level. The Systems Committee in their
recent survey found a clear need for a language or formalism on the conceptual description level, in order to describe in a consistent way the capabilities of various data management systems. Work at the physical and coded description levels is also needed in order to simplify the job of the data base manager and to make automatic data translation possible.

Questions:

4.1 Is there a need for data representation independence, i.e., to distinguish between logical and physical data?

4.2 If so, which levels of data description are needed to achieve it?

4.3 Is there a need for automatic data translation?

4.3.1 In converting from one machine to another

4.3.2 In computer networks in which data is shared among computers of various kinds

4.3.3 In providing compatibility among various data base management systems and procedural languages

4.3.4 In processing archival data recorded on magnetic tape and cards

4.4 Which levels of data description are needed in each of the cases cited in question 4.3?

4.4.1 Should data descriptions be stratified into four levels, as outlined above

4.4.2 Can work proceed on each level independently of the others

4.5 Is there a need for a common data description language?

4.6 Is the Data Division of COBOL a suitable base for a common data description language?

4.7 Should the efforts of the USASI committee working on data description languages be coordinated with CODASYL work?
Sketches

As a regular section of the SICFIDET Newsletter, we will publish short descriptions of work in progress, work proposed or interests in research in the areas of description or translation. To reduce some of the work involved for you and us, we are providing a form with, hopefully, enough questions to include what you have found relevant and to locate the areas you think need more investigation. This can become a very important part of the Newsletter, both for finding people working on similar problems and (hopefully) for help in forming working subcommittees, so please take the time to fill one out.

---

Bibliographic Information Request Form

Send to: Don Hatfield, IBM Cambridge Scientific Center, 545 Technology Square, Cambridge, Mass. 02139.

Name: ________________________________

Coordinates: ____________________________

Interests related to file definition and translation:

Related projects; post, current and future:

Publications:

Related professional activities:
Jay L. Cunningham  
Associate Specialist  
Bibliographic File Organization Project  
University of California, Berkeley.

Involved in a continuing study of computer based library systems, focusing on organization and search (terminal oriented) of large random accessed files. The data base will consist of bibliographic holdings records in machine form, and will be developed using both existing records and original input. This data base is open to standardization.

Len Swanson - Director of Operations  
Computing Center  
Beloit College  
Beloit, Wisconsin 53511

Has developed a general information system designed around the concept of a mnemonic dictionary specification of file and field names, and is interested in what other users have done in the area of file description.

Arnold Inselberg  
San Jose Research Laboratory  
Monterrey and Cottle Roads  
San Jose, California

Working on the manipulation of large files through a graphic interface, and the study and evaluation of linked data structures for large files.

Stan Rifkin  
13820 Victory Blvd.  
Suite 315  
Van Nuys, California 91401

Proposed doctoral thesis concerned with developing a function which maps hierarchical data structures into array (matrix) structures, and vice versa. Is concerned in particular with mapping FORTRAN arrays into COBOL hierarchies.
Paul Loewner  
IBM Research  
P. O. Box 218  
Yorktown Heights, New York 10598

Has developed a 'Federated Programming System', which allows for the linkage of different programs through data declaration and reformatting, equivalencing of synonyms. The system uses explicit data descriptions for omits and for variable typing.

JCR Licklider  
Director Project MAC M. I. T.  
545 Technology Square  
Cambridge Mass. 02139

An "on-line community" of research people who cooperatively build up a large shared information base of procedures and data.

Wants to achieve that objective, it is essential inter alia to associate with every sharable data set a formal description of its structure and representation, a description meaningful to people and to procedures.

Hopes to see the objective approached by Multies and other interactive computer systems that will be used in research in the Cambridge area during the next five years.
Publications

Considering the domain of SICFIDET to be strictly the systemization of description and translation rulers, there has been very little written that is substantial, largely because this is still a new area of investigation. Considering the interest beginning to be expressed in description schema we assume that this lack will not remain.

There are two books which can serve for good background reading on file structures and description systems. The first is Volume I of Knuth's The Art of Computer Programming, Addison Wesley, Reading, Mass., 1968, Chapter 2, "Information Structures", gives a good analysis of the various storage structures used commonly in programing, as well as a description of the more abstract mathematical objects they are used to represent.

There is some formal semantics and elegant analysis of some properties of trees and graphs. This is probably the most complete discussion available of data organization.

A second book with two articles directed precisely to the nature of description and the data described is the Annual Review in Automatic Programming, Vo. 5, Pergamon Press, N.Y. 1969, "Data Structures and their Representation in Storage," by M.E. D'Imperio, is an account of the comparison of a similar data structure implemented in several storage structures. "Generalized File Processing," by W.C. McGee, presents a comparison of nineteen file processing systems with respect to their file structures, the file processing operations which can be performed, and the processing language seen by the user. Included is a formalism for describing hierarchial data structures. Especially inasmuch as they give detailed comparison of many systems under one set of categories, both articles are valuable reading for anyone investigating the problems of general description systems.

As the study of file structures itself is, in general, necessary for their description, a good explanatory book in this area of data mangagement and information retrieval is File Structures for On-Line Systems, Spartan Press, N.Y., 1969, by professor David Lefkovitz. There is a comprehensive description of the translation of queries into access method calls, and a detailed overall development of the functional structure of IR systems.

There are many publications in this area of data management, information retrieval, scientific systems, programming languages, computer aided design and computer aided instruction, etc.... which contain some form of description of data and for storage structures. Few of these descriptions contain generality, few
are comparative. A recent exception is the CODASYL report, though it is hardly rigorous. When such exceptions continue to occur we will list and abstract them. In addition, there are articles which give the definitive description of a storage structure organization, an accessing technique or a data management principle. Some of these are seven years old and still instructive. In future issues of the Bulletin, we shall print lists of these articles by area, along with reviews of current work.

**Interesting Happenings**

**Mondays - MAC - SICFIDET Seminar**

2:30 - 5:00 Project MAC Conference Room 8th Floor  
545 Technology Square  
Cambridge, Mass.  
Open to the public.

27 August - SICFIDET Meeting at ACM National Conference  
San Francisco Hilton  
(See note inside)