DATA STRUCTURE LEVELS IN INFORMATION SYSTEMS

Kenneth J. Fogt

Systems Development Corporation

Prepared for:

Advanced Research Projects Agency

28 June 1973

TM-5123/000/00

DATA STRUCTURE LEVELS IN INFORMATION SYSTEMS

28 JUNE 1973

THIS REPORT WAS PRODUCED BY SYSTEM DEVELOPMENT CORPORATION IN PERFORMANCE OF CONTRACT DAHC15-73-C-0083, ARPA ORDER NO. 2254, PROGRAM CODE NO. 3D30.

SDC SYSTEM DEVELOPMENT CORPORATION
2500 COLORADO AVENUE ▪ SANTA MONICA, CALIF. 90406

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY *(Corporate author)*
System Development Corporation
2500 Colorado Avenue
Santa Monica, California 90406

2a. REPORT SECURITY CLASSIFICATION
Unclassified

2b. GROUP

3. REPORT TITLE
Data Structure Levels in Information Systems

4. DESCRIPTIVE NOTES *(Type of report and inclusive dates)*
Special Technical

5. AUTHOR(S) *(First name, middle initial, last name)*
Kenneth J. Fogt

6. REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
28 June 1973	15	13

8a. CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
DAHC15-73-C-0080	
b. PROJECT NO. 3D30	TM-5123/000/00
c.	9b. OTHER REPORT NO(S) *(Any other numbers that may be assigned this report)*
d.	

10. DISTRIBUTION STATEMENT
Approved for public release; distribution unlimited

11. SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
	Information Processing Techniques Office Advanced Research Projects Agency

13. ABSTRACT

This paper describes a categorization of data structure characteristics which are intrinsic to information system environments. An approach to transferring data bases among information systems based on these data structure levels is presented.

DD FORM 1 NOV 68 1473

UNCLASSIFIED

Security Classification

14 KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT

TM-5123/000/00

DATA STRUCTURE LEVELS IN INFORMATION SYSTEMS

28 JUNE 1973

K. J. FOGT

THIS REPORT WAS PRODUCED BY SYSTEM DEVELOPMENT CORPORATION IN PERFORMANCE OF CONTRACT DAHC15-73-C-0080, ARPA ORDER NO. 2254, PROGRAM CODE NO. 3D30.

ABSTRACT

This paper describes a categorization of data structure characteristics which are intrinsic to information system environments. An approach to transferring data bases among information systems based on these data structure levels is presented.

TABLE OF CONTENTS

1. INTRODUCTION

The need for the transfer of data bases among information systems grows constantly as more and more data is collected and manipulated by computers. The introduction of new, more advanced information management systems requires that data bases created and manipulated on old systems be reshaped and transplanted into equivalent data bases for new systems. (By "information management systems," we mean any computing environment supporting data base retrieval and generation.) This process is usually inefficient, slow, and expensive, especially for large data bases. The need for data base transfer also arises in situations where the data that exist in one application system could be useful for a second application system, provided that they could be restructured to be operated on by the second system.

The Common Information Structures project is attempting to develop a methodology and techniques for restructuring and transferring data bases across disparate information systems. Information system data structures, in their physical organizations and embodiments, reflect the uniqueness of the system applications. Each system employs data structure organizations, data access methods, and data management functions specifically tailored to a particular application. As a result, one information system cannot readily access data contained in another information system, particularly when the two systems function within different operating environments. Our goal is to make wider access possible by describing data bases in terms of data structure levels and transferring data bases from one system to another through these levels.

For the past year, in pursuit of this goal, we have studied numerous documents in the areas of data structure organization, information algebra, and functional properties of data management systems (see references). This provided the basis for characterizing information structures at three levels: logical, storage, and physical. Data structures at the logical level reflect information about elements in the data base, relationships among them, and the ordering on them. Data structures at the storage level reflect access paths and index organizations; these structures are used by system designers mostly to provide time/space efficiency (such as partially or fully inverted organization of data). Data structures at the physical level reflect file and record organization and are intimately connected to operating systems and their access methods.

This characterization of data structures forms the basis for an approach to data base transfer in which data are transformed from their physical representation to their storage representation to their logical representation in the source system. By means of common information structure techniques, this logical representation is transformed into the equivalent target logical representation for the system. Finally, the target logical data are transformed through the storage level to the physical-level representation in the target system. The techniques involving common information structures require a logical description of the source data base and a mapping of that description into an equivalent logical description of the target data base. This data base

description mapping serves as the basis for the specification of a data base transformer that takes instances of source data and transforms them into instances of target data.

2. DATA STRUCTURE LEVELS

We feel most data bases can be viewed as collections of descriptions about discrete entities which we call "individuals." These descriptions are instantiations of properties with property values that characterize individuals. We call these characterizing properties "data elements," the property values "data values," and the instantiated descriptions "data items."

We believe that the users of an information system should be given the capability to describe data bases at a logical level, independently of both the system's internal design and the characteristics of the computer hardware. Data structure characteristics that reflect logical entities and relationships among these entities, as viewed by a system user, constitute this logical data structure level. Such characteristics include:

(1) specifications of individuals and data elements
(2) specifications of type and range of data values
(3) specifications of data element relationships with respect to individuals
(4) specifications of data items
(5) grouping and ordering of data items
(6) specifications of maps and relationships between groups of data items

The storage data structure level comprises data characteristics which effect data inversions and indexing organization of the data. System designers manipulate storage data structure characteristics to achieve time/space efficiency. These characteristics may be peculiar to a specific information system and include:

(1) facility to create or modify data access paths
(2) specification of index organization
(3) specifications of inversions of data
(4) alternative ordering specifications
(5) realization of data items, groups, and orderings with specific path indicators (pointers)
(6) realization of relationships between groups of data item characters such as length and character set representation
(7) sequencing considerations like ISAM, HSAM, etc.

The physical data structure level comprises data characteristics that are intimately connected to operating systems. These characteristics include:

(1) realization of pointers by addressing techniques
(2) character set representations (octal, hex, etc.) and computer word specification like "byte," "bit," etc.
(3) blocking specifications for I/O purposes
(4) buffering considerations
(5) channel I/O
(6) specification of inter-record gaps
(7) device-dependent access specifications such as cylinder track considerations on disk, blocking factors on tape, or paging considerations on drum

Figure 1 shows what might be typical representations of data in the various data structure levels. At the logical level, the manner in which individuals are characterized by data items is shown. The storage level shows additional access and ordering paths which are realized through pointers. At the physical level, addressing techniques are used to realize the data organization. In this representation, the data items are linked by addresses and deciphered by length specifications.

Data structure characteristics that are internal to specific computing machines constitute the machine data structure level. These characteristics include device considerations and hardware constraints that prescribe interfaces between the operating system and the machine. The MITRE Report [12] on data management systems illustrates that machine-level data characteristics of different machines are so diverse that building data transformers for data representations at this level is both unreasonable and unnecessary. In our approach, we plan to take advantage of existing information system query and generate capabilities to read source and write target data bases. Consequently, the machine data structure level, per se, is not involved in the data transformation process.

The following list summarizes the typical characteristics of the various levels of data structures.

- Logical level
  - User's concept of the data base
  - Individuals
  - Data elements
  - Data values
  - Data items
  - Data element relationships and links
- Storage level
  - Data access and index organization
  - Keys, inversions, orderings
  - Time/space considerations

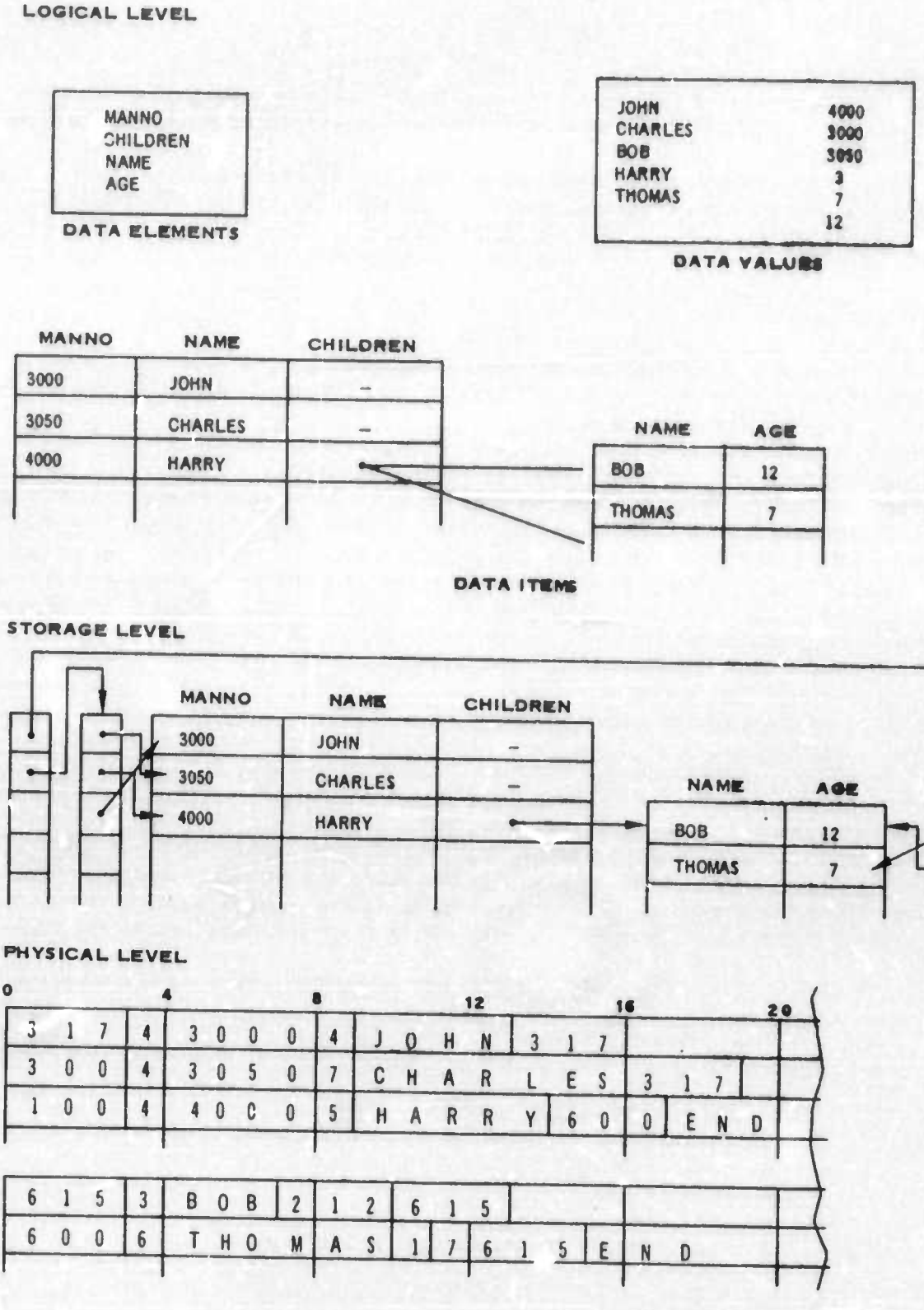


Figure 1. Data Transformation Between Data Structure Levels

- Physical level
    - File organization
    - Record specification
    - Operating system I/O
- Machine level
    - Device considerations
    - Hardware constraints
    - Intimate operating system/computer interface

3. IMPLICATIONS OF DATA STRUCTURE LEVELS FOR DATA TRANSFER

Figure 2 is an illustration of a typical data base transfer. For convenience and conciseness, we introduce the following notation.

- S_i represents the i^{th} information system in an environment of n systems ($2 \leq i \leq n$).
- LDDL(i) denotes the logical data description language of S_i.
- LDD(i,x) is a logical data description of a data base x in S_i expressed in the language LDDL(i).
- LD(i,x) denotes the logical data of data base x according to the description LDD(i,x).
- SDDL(i), SDD(i,x), SD(i,x), PDDL(i), PDD(i,x) and PD(i,x) are defined analogously for the storage and physical data structure levels, respectively.

We read data base x in S_i and extract PD(i,x). PDD(i,x), a specification of how the PD(i,x) is organized, is also created. A transformation is now performed on PD(i,x) to make SD(i,x). During this transformation, operating system (physical-level) data characteristics are removed. The SDD(i,x) created describes how the SD(i,x) is organized. The data transformer that extracts LD(i,x) from SD(i,x) removes information-system-dependent (storage-level) characteristics. If we restrict our attention to (or create) the rare case of information systems that have common logical data characteristics, then the transformation LD(i,x)→LD(j,x) is trivial. The usual case, however, will be handled by a data transformer that maps instances of LDD(i,x) to equivalent instances of LDD(j,x). Once LDD(j,x) and LD(j,x) are obtained, transformations will be performed which invoke necessary information system (storage-level) and operating system (physical-level) data characteristics so that the target physical data (PD(j,x)) will be an efficient representation for the target system S_j. The final step is to write PD(j,x) as data base x on S_j.

In summary, the data structure levels are used to modularize the data transformation process. Three data description languages are defined: logical,

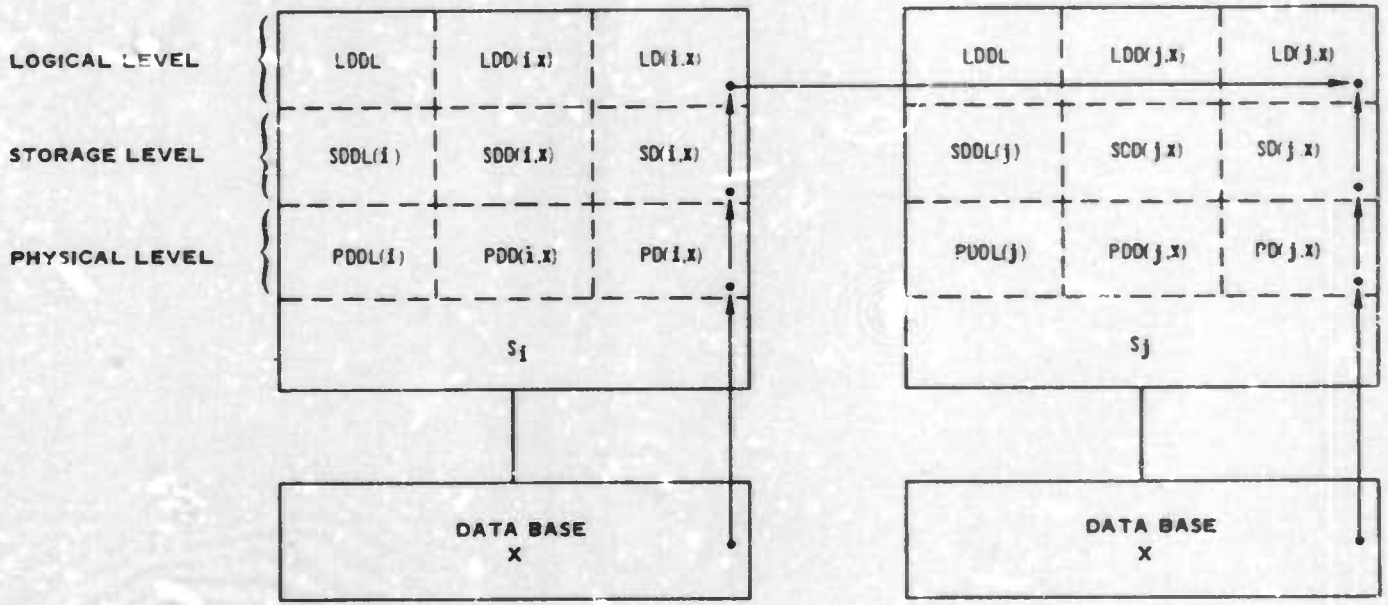


Figure 2. Data Base Transfer through Data Structure Levels

storage, and physical. Each is a formalism to describe the relationships between its corresponding data structures and data. Data are transformed from physical to storage, storage to logical, logical to storage, and storage to physical data structure levels by data transformers.

4. A SHORT-RANGE APPROACH TO DATA TRANSFER

Transferring data through the various data structure levels requires a preliminary data transformer design specification, a data description language specification, and implementation of data transformers for each level. Information system users are primarily interested in accessing data--not in the technical elegance of the access procedures. There are numerous situations in which a user might want to transfer an existing data base from one system to another only once: when his information system becomes obsolete and is replaced with an updated version, when he changes his hardware configuration, when he wishes to change information systems for efficiency considerations, when he wishes to have his data accessible on a different manufacturer's machine, etc. It would be unreasonable in these situations to put a large effort into automating transformers or designing new data structures for his application, since he will only do it once. Consequently, we are addressing data base transfer in an evolutionary manner. We are basing our transfer process on logical level techniques. We will transform a source data base from the information system environment to the logical level by "hand-tailored" mechanisms. We will then transfer this logical data from source system to target system with logical data structure techniques. The target system logical data will be written into the target system environment in an expedient manner.

These techniques can then be refined and embellished in a longer-range plan where the transformation between levels is effected in an efficient and elegant manner, taking full advantage of data characteristics at the three data structure levels.

We introduce a few additional notations to precisely express the tasks involved:

- FDL(i) represents the file description mechanism (as it exists) in S_i.
- LDDL(i) denotes the logical subset of FDL(i) which is relevant to the data transfer process.
- The set of all possible data base descriptions LDD(i,x) in system S_i, is denoted LDD(i).
- A data description mapping, which specifies for every instance of LDD(i) an equivalent instance of LDD(j), is called M_{ij}.
- A data transformer, LD(i,x)→LD(j,x), is denoted $T_{ij}(x)$.

We envision the data transfer process to begin with the derivation of an LDD(i,x) in the language LDDL(i). The data description mapping M_{ij} will then specify an equivalent LDD(j,x) given this LDD(i,x). The logical data LD(i,x) will be extracted from data base x using the query capabilities of S_i and the DD(i,x). The data transformer $T_{ij}(x)$ will transform instances of LD(i,x) to instances of LD(j,x), which will then be written in S_j using the S_j file generate capabilities and the LDD(j,x). The specific tasks and our approaches to them are outlined below.

4.1 ANALYSIS OF FDLs

In order to gain insight into logical data structures, we will analyze existing systems' file description languages (FDLs) and logical data structures as implied by those languages. We will concentrate on systems that are representative of diverse application areas in both their logical data structures and their internal structure characteristics. The systems that we plan to study include: DS/3--an SDC hierarchical data management system; IMS--a widely used IBM system that allows network data structures in addition to hierarchies; and the Datacomputer--the ARPANET data management facility.

4.2 DEFINITION OF LDDLs

We expect that many details expressed in the FDLs are reflections of physical characteristics of the information systems that do not need to be reflected in the LDDLs. For each system S_i, therefore, we will isolate or define its LDDL(i) from the file description mechanisms expressed in its FDL.

4.3 ANALYSIS OF LDDL CHARACTERISTICS

We will analyze the different LDDLs in order to isolate logical structure characteristics such as hierarchical or relational structures, depth of hierarchical levels, and size of relations. The remaining LDDL characteristics, such as the type, size, and format of fields, are data dependent. This separation of logical structures and data-dependent properties of LDDLs is important because the logical structures are the basis on which to define the mappings of LDDLs, and the data dependent properties are necessary for the creation of an efficient data base organization in the target system.

4.4 CLASSIFICATION OF DATA STRUCTURE COMPONENTS

Different existing systems' logical data structures need to be classified with regard to such things as terminology (e.g., "table" and "relation" are sometimes used synonymously), type (e.g., numeric, alphabetic), and intrinsic constraints (e.g., levels of hierarchies). This information is vital in order to recognize the commonality and differences among various systems' logical data structures.

For example, the CODASYL Task Group [7] attempts to allow COBOL system users to share data bases by "tailoring" a data base substructure for each user wishing

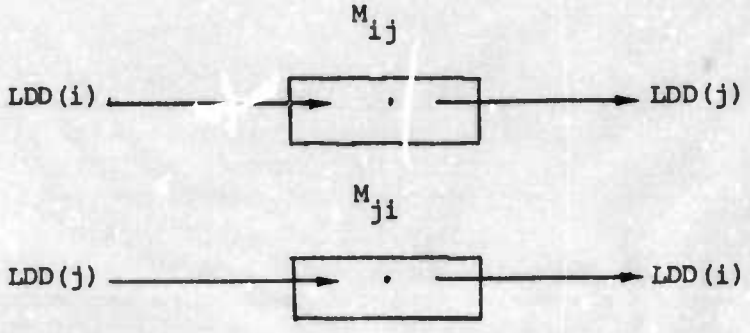
to access the data. The schema and subschema in this approach are forms of logical data descriptions. Terms like "groups," "arrays," "plexes," "trees," etc., are used to describe data structures. DIAM [4,5] uses "A-strings," "E-strings," and "L-strings" to describe similar data structures.

We will correlate the terminologies of the various systems so as to minimize the number of classifications of logical data structures as well as utilize the underlying commonality in our specification processes. The isolation of types of data structures will be used to determine the feasibility of mapping logical structures between systems. Intrinsic constraints like "levels" of hierarchy have serious implications for the complexity of the data transfer process. For example, suppose that there exists a data description mapping M_{ij} that maps LDD(i)→LDD(j). Further, suppose that systems S_i and S_j allow hierarchical data structures of depth 5 and 10, respectively. Then in the case of the need to transfer a data base with 10 levels on S_j into a combination of hierarchies of 5 levels or less on S_i, we need to isolate the logical relationship portion of the LDD(i) and, working within this framework, map the hierarchical structures of S_j into equivalent ones on S_i.

4.5 DATA DESCRIPTION MAPPING SPECIFICATION

Two alternative approaches to specifying data description mapping will be considered:

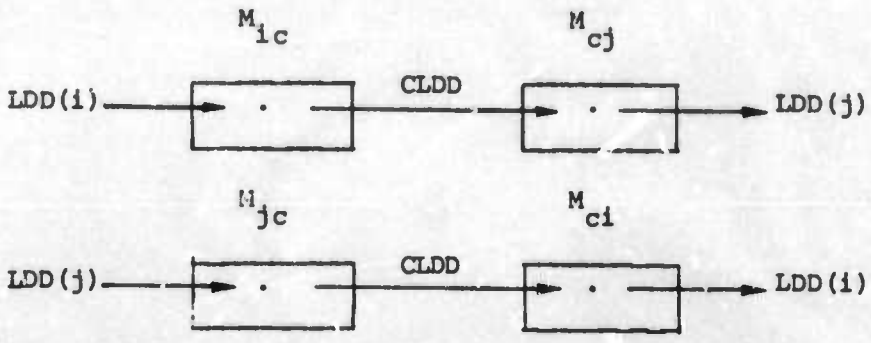
(a) Let S_i and S_j represent the i^{th} and j^{th} systems ($i \neq j$) in an environment of n information systems. There are n(n-1) mappings (M_{ij}) necessary in order to map every LDD(i) to every LDD(j). Two mappings are necessary for every pair i,j:



In most instances, M_{ij} is not the same as M_{ji}, since data structure types and constraints will differ between the systems. However, we should be able to take advantage of any commonalities found (from the studies in previous steps) in the generation of the M_{ij}'s.

(b) An alternative (and probably more efficient) way to achieve mappings between the data descriptions is by identifying the commonality instrinsic to logical data structures and logical data descriptions and defining a common logical data description (CLDD) and its corresponding language (CLDDL). We will explore the possibility of making the CLDDL some set-theoretic formulation of all the DDLs.

Let M_{ic} be a data description mapping from LDD(i) to a CLDD. Two typical mappings are necessary for systems S_i and S_j:



In this approach, only 2n mappings are necessary. However, this method requires specification of a CLDDL, whereas the first approach requires no such preliminary step. The more commonality we find in the logical data structures, the more efficient the second approach is.

4.6 DATA BASE TRANSFER

For the purpose of testing the ideas and techniques for data transfer, a preliminary experiment is necessary. We plan to extract LD(i,x) from the data base x in S_i with hand-tailored mechanisms based on S_i's query capability and the data base description LDD(i,x). We will build the data base transformer $T_{ij}(x)$, which transforms LD(i,x) into LD(j,x) based on the LDD(i)→LDD(j) specifications. $T_{ij}(x)$ will then transform LD(i,x) into LD(j,x), which will be written into S_j with S_j's file generation capabilities and the LDD(j,x) by similar methods. Possibly more elegant techniques could be explored at a later time, after the basic information transfer techniques are achieved.

5. EXTENSIONS

We envision three directions which could lead to more automatic and efficient techniques for achieving data transfer.

1. Automatic generation of data description maps. We expect that while we define specific mappings, we will find tools that can be realized by meta-compiler techniques. This process could lead to the automatic generation of data description maps. In particular, we believe that the common logical data description (CLDD) concept will be useful.

2. Automatic extraction and generation of data bases. This includes the development of a methodology for extracting data from and inserting data into data bases using data descriptions. We expect to take advantage of existing generate and retrieval mechanisms in current information systems. Research in this direction could lead to a study of levels of data structures within information systems, and their use in the process of transforming physical data into logical data.

3. Automatic transformation of data bases using the data description maps. This includes the development of more efficient methods of manipulating data during the transfer process, such as (1) buffering and (2) control of data flow from the source system to the target system.

6. REFERENCES

[1] Fry, J. P., University of Michigan, Smith, D. P., University of Utah, and Taylor, R. W., University of Massachusetts: "An Approach to Stored Data Definition and Translation," 1972 ACM SIGFIDET Workshop, pp. 13-57, December 1972.

[2] Fry, J. P., Frank, L. and Hershey III, E. A., University of Michigan: "A Developmental Model for Data Translation," 1972 ACM SIGFIDET Workshop, pp. 77-107, December 1972.

[3] Smith, D. P., University of Utah: "A Method for Data Translation Using the Stored Data Definition and Translation Task Group Languages," 1972 ACM SIGFIDET Workshop, pp. 107-125, December 1972.

[4] Astrahan, M. M., Altman, E. B., Fehder, P. L. and Senko, M. E. IBM: "Concepts of a Data Independent Accessing Model," 1972 ACM SIGFIDET Workshop, pp. 349-363, December 1972.

[5] Astrahan, M. M., Altman, E. B., Fehder, P. L., and Senko, M. E., IBM: Specifications in a Data Independent Accessing Model," 1972 ACM SIGFIDET Workshop, pp. 363-382, December 1972.

[6] Taylor, R. W., University of Massachusetts: "Generalized Data Base Management System Data Structures and Their Mapping to Physical Storage," Ph.D Dissertation, 1971.

[7] CODASYL Data Base Task Group Report, April 1971

[8] Senko, M. E., Altman, E. B., Astrahan, M. M., and Fehder, P. L., IBM: "Data Structures and Accessing in Data Base Systems," IBM Systems Journal, pp. 30-93, Vol. 12, No. 1, 1973.

[9] Language Structure Group of the CODASYL Development Committee: "An Information Algebra," Communications of the ACM, pp. 190-204, April 1962.

[10] Engles, R. W., IBM: "A Tutorial on Data Base Organization," Annual Review of Automated Programming, Vol. 7, Part 1, 1972.

[11] Huang, J. C., University of Houston: "An Algebraic Model for Data Base Management Systems," June 1971.

[12] MITRE Corporation: "Data Management Systems Survey," January 1969.

[13] EDP Analyzer: "The Debate on Data Base Systems," March 1972, Vol. 10, No. 3.