Annals of Economic and Social Measurement, 5/1, 1976

PROGRAMMING SOFTWARE NOTES

INFORMATION SYSTEMS FOR PUBLIC SECTOR MANAGEMENT*

BY WILLIAM D. HASEMAN AND ANDREW B. WHINSTON

This paper presents a generalized framework for managing and analyzing large economic data bases. The data bases have the common properties of (a) a complex structure, (b) a need for editing, and (c) repeated operations. A generalized data management system is described which is then interfaced to a Query Language Processor. The data cycle is then discussed with its relationship to the data merging and conversion problem. The framework is then applied to two different and specific studies. The conclusion of the paper is this framework which separates the data management functions from the economic analysis functions. This division of labor provides for a more consistent and useful system.

1. INTRODUCTION

During the last fifteen years, as the capabilities of computers have increased, we have witnessed a phenomenal growth in their use for policy analysis. A concurrent, if not spectacular, growth has taken place in the business management field to such an extent that the term Management Information Systems has been coined. The naming of the field is probably more of an indication of level of activity rather than the existence of an organized body of theory and practice. However, there has been an attempt by computer manufacturers and several software vendors to make available to companies a variety of software packages that would help in the development and operation of such systems. In fact, this need has developed a commercial market for what are referred to as File Management Systems or data management systems. These systems are valuable in that they both require the user to adapt certain formalization of the data management process and provide already developed software needed to carry out much of the development of the information systems.

This paper will explore the problem of data management in the public non-commercial sector of our economy. By attempting to formalize our approach to this problem, we can hope to understand the overlap with the problems of management information systems and of course, the differences. The ultimate value of a formal approach to this problem is that we may learn what aspects of software development are specific to any application and what parts are common to the typical implementation. The ability to conceptually introduce this division of labor should lead to the development of software which will expedite the development of public data systems. This paper gives views both at the conceptual and software level.

We first present a general framework for an information system. This framework underlines an implementation of a data management system developed in the field of water pollution control. We will also show that our

* This work was supported in part by Grant Number GJ377-55 from the Office of Computing Activities of the National Science Foundation and Office of Water Resources Research.

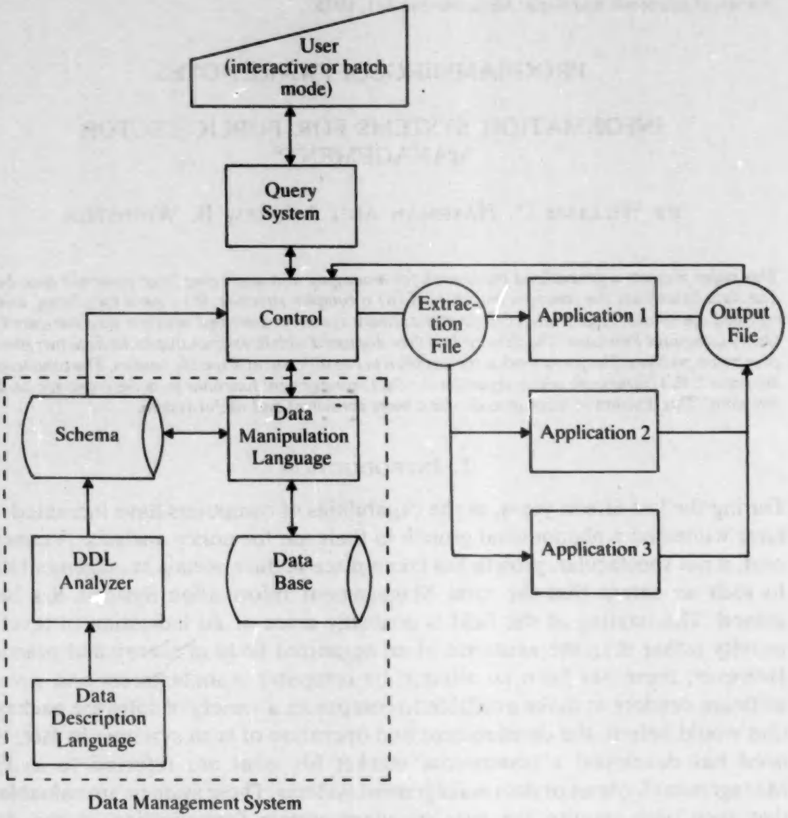

Figure 1 GPLAN System

framework is consistent with an extensive specific study of a different problem carried out by David, Gates, and Miller. [1].

Most of the studies of the nature under discussion begin with first determining the ultimate goal of the study, and then proceeding to search for existing data which can be used to accomplish this goal. Rausser and Howitt [2] presented a model which they developed for the stochastic control of environmental externalities, in which such data requirements were enormous. Although some projects have the luxury of collecting and verifying the specific data required for the study, the majority of the studies rely on tapes of data collected from numerous and unrelated previous studies. The tapes often present the researcher with overlapping and inconsistent data and with missing and unwanted data. Many large projects devote the majority of their effort in converting their initial data into a clean data base for their analysis. Before looking at the problem of developing a clean data base, it is first necessary to present a background discussion of data management systems and a general framework of a planning system we have developed around a data management system.

2. Data Management System

The structure of the GPLAN-Data Management System (GPLAN/DMS) is shown in Figure 1. Although only a brief discussion of this system will be included here, a more detailed discussion may be found in [3] and [4]. The GPLAN/DMS system is based on the specifications of the 1971 CODASYL Data Base Task Group (DBTG) report [5]. The data management system consists of the data base, a schema which is the logical description of the data base, and the data manipulation language (DML). The schema is generated from the Data Description Language (DDL) and contains a description of the record-types in the data base, the item-types contained in those record-types, and the set relationships formed by those record-types.

The user generates a schema by writing the data description in the Data Description Language, which is then processed by the DDL analyzer. An example DDL is shown in Figure 2, and the record occurrences associated with that DDL is shown at the bottom of the Figure. This DDL defines a simple water pollution data base which consists of three record-types and three sets. As can be seen, the first record-type ('BASN') contains three item-types ('BSID', 'BSNA', 'E'), and can be a member record in the set 'ALLB' and an owner record of the set 'RVIB'. The DDL for the total Data Base is generated only once, and the system saves this description in the schema for future use by the Data Manipulation Language (DML).

Since the DDL provides for a set structure relationship among the various record types, the system can store any logical structure ranging from simple sequential files to the most complex of networks. The examples shown in Figure 3 demonstrate three of the many possible data structures which can be stored in the data base. All of those examples assume that some economic data has been collected and that this information will be keyed on CITY, STATE, and COUNTRY.

The first example (3A) shows how the data could be stored as a sequential file. Each record occurrence is stored in this single set and can only be reached by sequentially searching through the set until the desired city is located. The second example (3B) demonstrates how the same data could be a structure in a tree-like structure. To locate a particular record occurrence, the user first selects the appropriate country, then the state, and then searches for the city of interest. The third structure (3C) is a network structure in which the owner member relationship is considerably more complex. This structure provides the user with the freedom of moving in any direction, but forces him to be concerned with such problems as looping. As can be seen by these three examples, the user can select the appropriate data structure which will satisfy his data requirements.

The Data Manipulation Language (DML) provides the user with the commands for manipulating records within the set structure and for storing and fetching data from those records. The DML consists of 53 commands which are shown in Figure 4. These commands appear as subroutine calls in the users'

	NAME	TYPE	SIZE	DEPEND	MAX	COMMENT
RECORD	BASN					BASIN
ITEM	BSID	INTEG	1		1	BASIN ID
ITEM	BSNA	CHAR	20		1	BASIN NAME
ITEM	E	REAL	1		1	MEAN ELEVATION
RECORD	RIVR					RIVER
ITEM	RVID	INTEG	1		1	RIVER ID
ITEM	RVNA	CHAR	20		1	RIVER NAME
RECORD	RECH					RIVER REACH
ITEM	RCID	INTEG	1		1	REACH ID
ITEM	RCHN	CHAR	20		1	REACH NAME
ITEM	DIST	REAL	1		1	LENGTH IN MILES
ITEM	DGO	REAL	1		1	DO GOAL
ITEM	NCON	INTEG	1		1	NUMBER MINERALS
ITEM	CGO	REAL	1	NCON	8	MINERAL GOALS

	NAME	ORDER	KEY	COMMENTS
SET	ALLB	SORT	BSNA	ALL BASINS
OWNER	SYST			SYSTEM RECORD
MEMBER	BASN			BASIN RECORD
SET	RVIB	SORT	RVNA	RIVERS IN BASIN
OWNER	BASN			BASIN RECORD
MEMBER	RIVR			RIVER RECORD
SET	RCRV	SORT	RCHN	REACHES IN RIVER
OWNER	RIVR			RIVER RECORDS
MEMBER	RECH			REACH RECORD

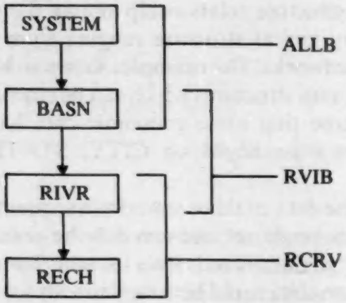

Figure 2 DDL Example

program. The programming languages which can accommodate these DML calls include FORTRAN, COBOL, PL/I, and assembly languages. Two examples of how the DML would look in a FORTRAN example are shown in Figure 5. These examples assume that the data base has the structure described in Figure 2, and as can be seen by these examples, the user is required to generate considerable code in order to be able to use the complex data structures. The next section will discuss the GPLAN framework which eliminates this effort from the user, and provides for a query capability for the data base.

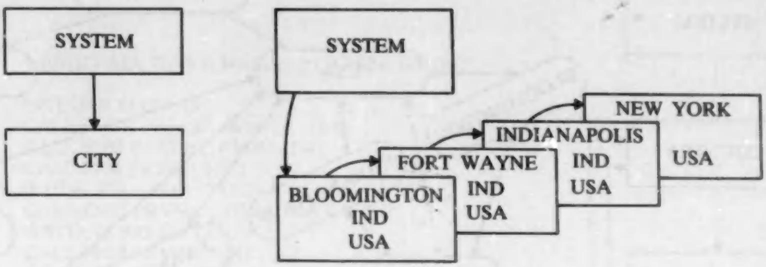

Figure 3A Sequential File

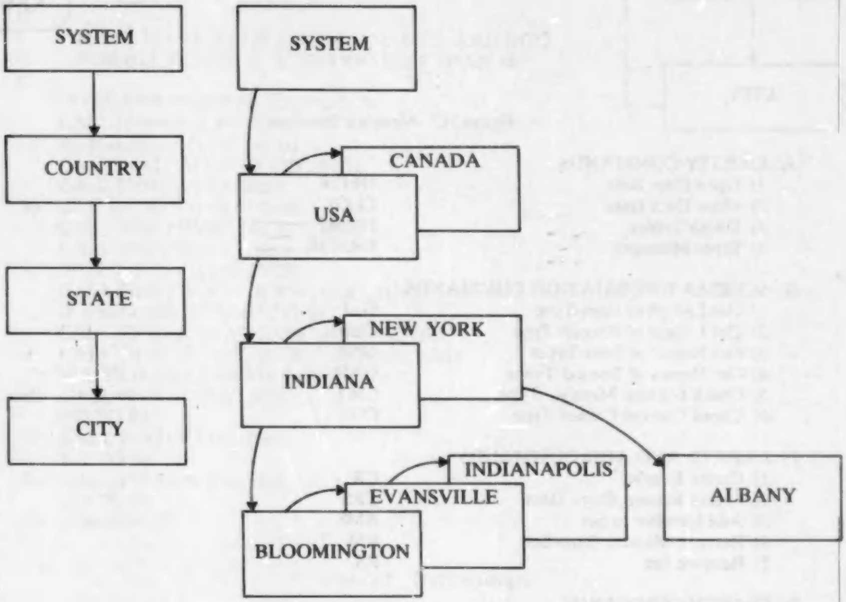

Figure 3B Tree Structure

3. GPLAN Framework

The Generalized Planning System (GPLAN) framework [6] [7] shown in Figure 1 was designed around the data management system to provide the user with as much isolation from the data and application programs as possible. The goal of the GPLAN system is to let the system handle the data problems, such as what format the data is in, and let the user concentrate on his goal of analyzing the data. With this in mind, the control program takes the request from the user in the form of a query, converts this to the appropriate DML and program calls, and generates the desired response. The control program, which uses Artificial Intelligence techniques [8], has access to a description of all the data in the data base, as well as a description of all functions or programs which operate on this data base.

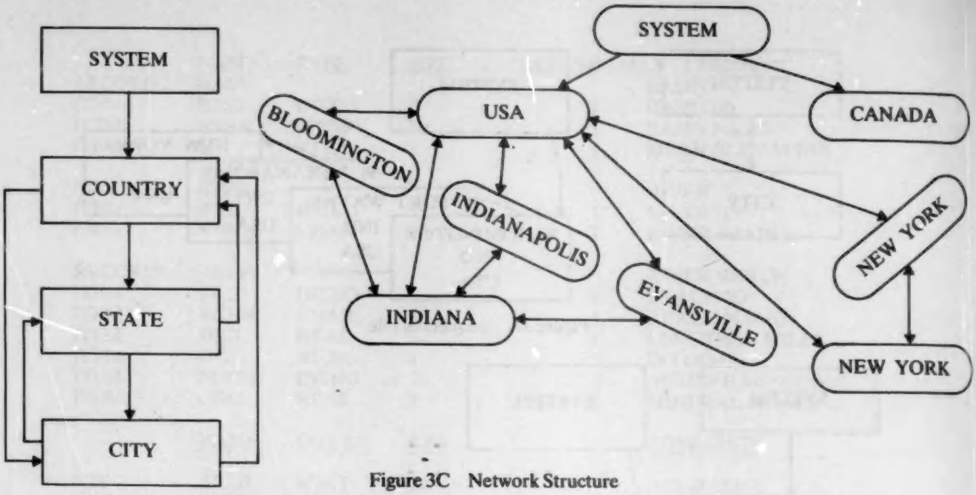

Figure 3C Network Structure

A. UTILITY COMMANDS

1) Open Data Base	OPEN
2) Close Data Base	CLOS
3) Dump Tables	DUMP
4) Error Messages	ERROR

B. SCHEMA INFORMATION COMMANDS

1) Get Length of Item-Type	GLI
2) Get Length of Record-Type	GLR
3) Get Names of Item-Types	GNI
4) Get Names of Record-Types	GNR
5) Check Current Member-Type	CMT
6) Check Current Owner-Type	COT

C. CREATE AND ADD COMMANDS

1) Create Record	CR
2) Create Record/Store Data	CRS
3) Add Member to Set	AMS
4) Remove Member from Set	RM
5) Remove Set	RS

D. SEARCH COMMANDS

1) Find First Member	FFM
2) Find Last Member	FLM
3) Find Next Member	FNM
4) Find Previous Member	FPM
5) Find Member Based on Sort Key	FMSK

E. DATA MANIPULATION COMMANDS

	KEY	CURRENT MEMBER	CURRENT OWNER	CURRENT RECORD
1) Delete Record	DRK	DRM	DRO	DRR
2) Get Data	GETK	GETM	GETO	GETR
3) Get Field	GFK	GFM	GFO	GFR
4) Get Key		GKM	GKO	GKR
5) Get Record Type	GTK	GTM	GTO	
6) Set Field	SFK	SFM	SFO	SFR
7) Set Current Member	SMK	SMM	SMO	SMR
8) Set Current Owner	SOK	SOM	SOO	SOR
9) Set Current Record	SRK	SRM	SRO	

Figure 4 DML Commands

```
C
C
C      PRINT ALL RIVER NAMES IN BASIN ‘WHITE’
C
       INTEGER DATA (5)
       CALL FMSK (‘ALLB’, ‘WHITE’, IER)
       CALL SOM (‘ALLB’, ‘RVIB’, IER)
       CALL FFM (‘RVIB’, IER)
10     IF (IER. EQ. – 1) GO TO 20
       CALL GFM (‘RVNA’, ‘RVIB’, DATA, IER)
       WRITE (6,100) DATA
       CALL FNM (‘RVIB’, IER)
       GO TO 10
20     CONTINUE

C
C      PRINT RIVER NAMES, REACH NAMES, AND DGO
C      FOR ALL REACHES WITH DGO LESS THAN 10
C
       INTEGER RVNAM (5), RCNAM (5)
       CALL FFM (‘ALLB’, IER)
10     IF (IER. EQ. – 1) GO TO 90
       CALL SOM (‘ALLB’, ‘RVIB’, IER)
       CALL FFM (‘RVIB’, IER)
30     IF (IER. EQ. – 1) GO TO 80
       CALL SOM (‘RVIB’, ‘RCRV’, IER)
       CALL FFM (‘RCRV’, IER)
50     IF (IER. EQ. – 1) GO TO 70
       CALL GFM (‘DGO’, ‘RCRV’, DGO, IER)
       IF (DGO. GE. 10.0) GO TO 60
       CALL GFM (‘RVNA’, ‘RVIB’, RVNAM, IER)
       CALL GFM (‘RCHN’, ‘RCRV’, RCNAM, IER)
       WRITE (6,200) RVNAM, RCNAM, DGO
60     CALL FNM (‘RCRV’, IER)
       CO TO 50
70     CALL FNM (‘RVIB’, IER)
       GO TO 30
80     CALL FNM (‘ALLB’, IER)
       GO TO 10
90     CONTINUE
```

Figure 5 DML Example

The basic query language for the GPLAN framework consists of three parts:

⟨COMMAND⟩⟨VARIABLE clause⟩⟨CONDITION clause⟩

The COMMAND tells the query processor which action is to be taken on the data involved. The commands which are currently implemented include:

1) LIST	List the desired data
2) FIND	Find the desired data
3) PLOT	Plot the desired data
4) REGRESS	Regress the desired data
5) CHANGE	Change the desired data
6) ADD	Add a new record occurrence
7) DELETE	Delete a record occurrence
8) RUN	Execute a users’ program

The variable clause describes the data items involved in the query and consists of item-types and the following operators (+, –, *, /, **, sin, cos, log, etc). For example:

LIST NAME WAGE * HOURS-WORKED

would list each NAME and the value of WAGE times HOURS-WORKED. The condition clause permits the user to selectively retrieve data from the data base. The condition clause includes all the operators in the variable clause plus the logical operators (=, ≠, <, >, ≤, ≥, AND, OR, NOT). The following two example queries would produce the same results as the two examples of DML shown in Figure 5.

LIST, RVNA FOR BSNA = 'WHITE'
LIST RVNA, RCHN, DGO FOR DGO < 10

As can be seen, the query language can eliminate a lot of programming effort which would normally be invested in writing DML.

As the query language is developed further, the user will become even less involved with which Data Manipulation programs that are actually being used to generate the desired results. When the control program determines that a particular module is required to answer a question, it will generate the appropriate input file for that program from the data base. Using this structure, the user will not have to rewrite his existing programs and does not have to learn how to use the DML commands. This also means that if the data base was to be restructured, no changes would be required for any of the users' program or for the control program. This restructuring capability is important for the discussion which follows.

Before looking at how the GPLAN system can be useful in large scale economic data bases, it should be emphasized again that the GPLAN framework is a generalized software structure built around a conventional data management system for the purposes of minimizing the users' efforts in the area of data manipulation. It should be noted that for most projects, even the data management system would be a substantial improvement over the current state of the art in this area of setting up large scale economic data bases.

4. Data Cycle

The data cycle for a particular economic data study could be viewed as shown in Figure 6. The data is initially collected and stored either on cards or most likely on a tape. This data will be referred to as the "raw data," because before this data can be of benefit to the user, it must go through a filter system which cleans, verifies, and aggregates the data. Once the data is passed through this filter, it is considered "clean data" and then can be analyzed by the various programs involved. After a period of time, the data may become out of date and it is then passed through another filter to be stored in the archives, which is generally a tape. This information should be saved, since it might have to be reloaded back into the data base at some later point in time.

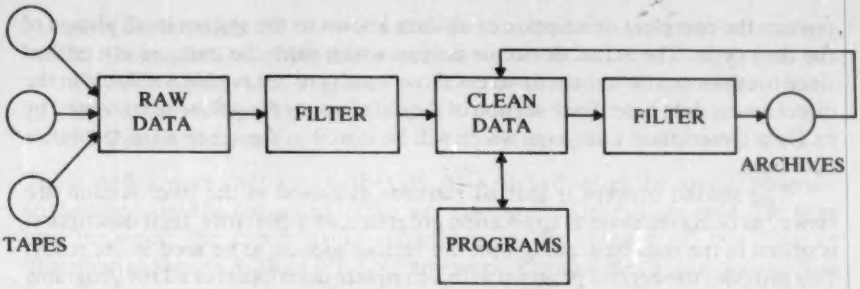

Figure 6 Data Cycle

Before looking at the appropriate data structures for each phase in the data cycle, some discussion will be devoted to the processes which are involved in the two "filter" boxes. The first filter should perform the following:

a) Verify existing data using either known bounds, or by comparing equivalent data from two sources, or by comparing the data to known relationships among other data items.
b) Replace missing data whenever possible from alternate sources or from some estimation procedure such as regression.
c) Aggregate data whenever required.
d) Provide a mapping of this modified "raw data" into the "clean data" structure.

Several studies have been directed towards solving the problems in a, b, and c; for example, see Alter [9], Marsden [10], and Ruggles [11]. In the framework, we are proposing these "filters" would merely be application programs which would be executed by the user through the query language. The data management system itself can be used to help detect missing and erroneous data since the user could specify a range for each data value. The question of a mapping will be performed by the control program, once the two data structures ("raw" and "clean") are defined. This process will be discussed in more detail in the next section.

The second filter is used to further aggregate the data for storage in the archives. The structure of the archive data should be such that it can easily be reloaded back into the data base for further studies. The problem of data verification should be resolved by the time the data is ready for the archives.

5. Data Structures

With the background of the data management system, the GPLAN structure, and the data cycle, the question to be addressed is how to structure the data base for any given application. The first concept in describing our approach to solving this problem is that since that data management system can support a vast number of data structures, then all phases of the data cycle can be viewed as being in the same data base. By using this conceptual viewpoint, the control program will possess the complete description of all data known to the system in all phases of the data cycle. The actual device or devices which store the data are not critical since routines can be written to access those which are not readily available in the direct access data base. Each section of the data base or file will be represented by its Data Description Language which will be stored in the schema for the entire data base.

The second concept is that all routines discussed in the filter section are viewed as being the same as application programs, and therefore, their description is stored in the data base along with the various models to be used in the study. This provides the control program with a complete description of all the programs in the system as well as all the data known to the system. Included in this group of programs would be generalized data manipulation routines such as plotting, regressions, histograms, and statistics.

The only missing link in this process is the data which results from the various analysis to be performed. It would only seem logical that this data also be stored in the data base so that it will be available for other future analysis as well as for storing it in the archives.

The various programs are viewed by the control program as merely being functions which move data from one section of the data base to another. The logical structure for each of these sections of the data base should be determined by the queries requested by the user, which in turn, requires the use of one or several of the application programs available. Using this conceptual structure, as the GPLAN system currently does, the various sections of the data base will expand and contract dynamically as the requests from the user will change from time to time.

This adaptive type of data structure will be transparent to the user and will provide the maximum amount of flexibility for the control program to schedule the execution of the various functions while hopefully trying to minimize the use of resources. As time goes on, and the user adds more and more application programs to the system, the capabilities of the control program will increase, and the user will be required to perform even less of the remedial data handling tasks.

6. Water Pollution Example

The GPLAN system is currently being used in the area of water pollution control for the Indiana Stream Pollution Control Board. One of the particular problems solved using this system was the determination of a waste treatment program which would satisfy Section 303 of the 1972 Water Quality Act for the Grand Calumet River Basin in Indiana. The specific problem required developing a sewage treatment program which would meet the specified water quality standards while at the same time requiring the minimal investment in additional facilities. The solution required data from four different sources, and the use of five models written by the user. The data was collected and stored in a hierarchical data structure which is similar to the structure shown in Figure 2.

The models developed for the study were the following application programs:

A) Cost Analysis
B) Determination of River Parameters
C) River Simulation Program
D) Nonlinear Optimization Routine
E) Report Generation

These models were interfaced to the GPLAN system through the control program so that they could be requested by the user through the Query Language. The user also possessed the capability of requesting specific queries of data base as described in Section 3. This GPLAN framework provided the capability of getting this system up and working with a minimal amount of effort. The generalized data management system also provided the user with considerable data manipulation and editing capability which would most likely be absent from a specific data base.

7. Economic Data Example

An indepth study, described in *Linkage and Retrieval of Microeconomic Data* by David, Gates, and Miller [1] is concerned with data collected for the Wisconsin Assets and Income Studies (WAIS). The data collected was from the following sources:

1) Master Tax Record File
2) Property Income File
3) Social Security Benefit File
4) Social Security Earnings File
5) Personal Interview Survey and Assets Diary
6) Identification File
7) State Tax Pool

The study first presents an excellent discussion of the data, its characteristics, and its structure. The book then takes a look at a specific data structure which involves a linked structure of records. After the data structure is defined, a series of error detection and correction techniques are discussed as they relate to the specific data structure. Each of the analysis programs which are used with the data base are designed to handle the specific linked data structure developed by the authors. The last chapter of the study briefly discusses the similarities of their system to a Management Information System.

The real problem with this particular study and with many economic studies was that the authors attempted to mix the design of an information system with the study of economic data. The result was a specialized system which possesses very little flexibility and adaptability to future studies. The work this study provided, as far as the economic study was concerned, was quite contributive, however the data capabilities are fairly primitive when compared to a generalized data management system. Many of the programs developed for merging data and for error checking were designed for the specific structure and will be of little use in future studies. This problem could have been avoided had these programs been interfaced with a generalized data management system.

In order to illustrate this point, we can look at one of the specific data structures discussed in the text. The logical structure as shown in Figure 7, is from

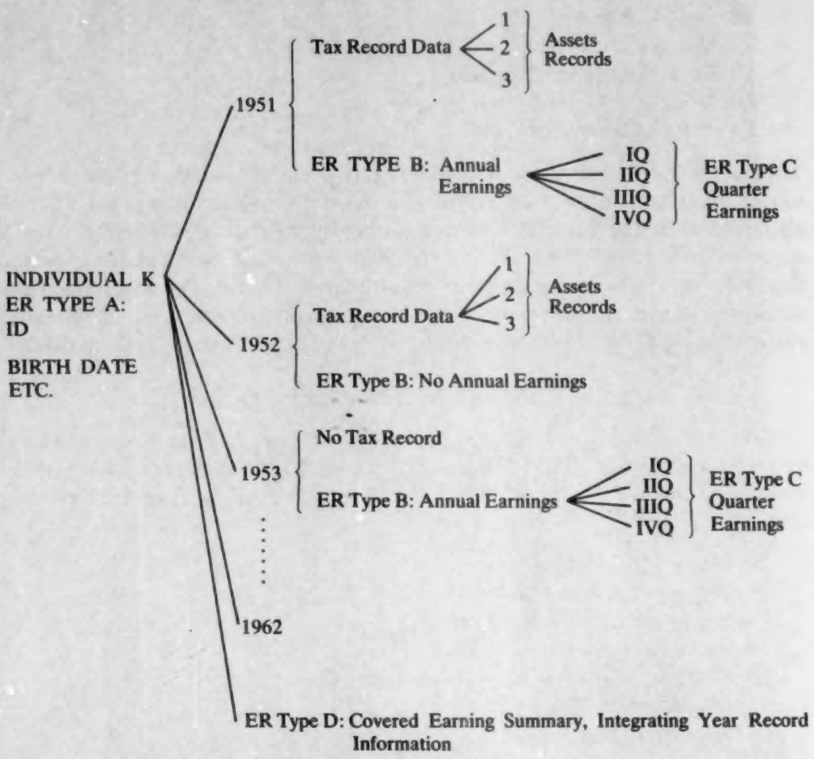
INDIVIDUAL K
ER TYPE A:
ID
BIRTH DATE
ETC.
1951
Tax Record Data
1
2
3
Assets Records
ER TYPE B: Annual Earnings
IQ
IIQ
IIIQ
IVQ
ER Type C Quarter Earnings
1952
Tax Record Data
1
2
3
Assets Records
ER Type B: No Annual Earnings
1953
No Tax Record
ER Type B: Annual Earnings
IQ
IIQ
IIIQ
IVQ
ER Type C Quarter Earnings
1962
ER Type D: Covered Earning Summary, Integrating Year Record Information

Figure 7 Logic of Tax and Earnings Records Data Structure

page 91 of the text, and is the logical structure of the Tax and Earning Records data. The study proposed trying to represent this structure by having several sequential files which would contain pointers in one file which would correspond to a related record in another file. It is not clear from the study how much of this interrelationship by pointers was actually implemented, because of the obvious difficulty with the required bookkeeping.

This problem of using pointers to interrelate data could all be handled by a generalized data management system, and the user would not have to keep track of the linkages himself. This particular structure could easily be realized using the GPLAN/DMS with the actual data structure as shown in Figure 8. If each of the other nine files discussed in the study were represented in a logical structure as shown in Figure 8, then the entire data could be stored in one logically consistent data base, and the user would be able to define all the interrelationships which actually existed between all the data. This would also provide greater flexibility for accessing the data for the future economic studies.

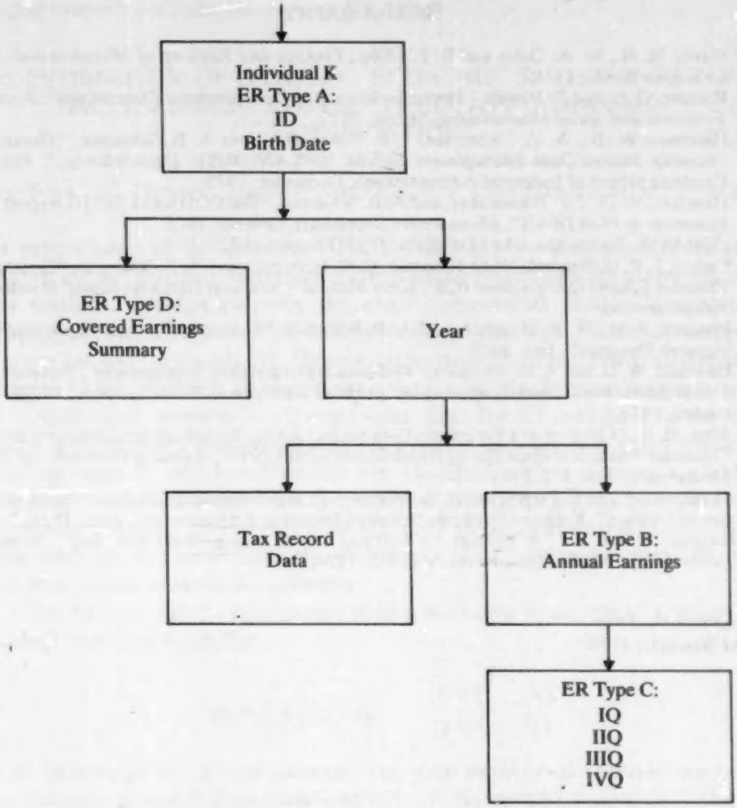

Figure 8 GPLAN Data Structure for Tax and Earnings Records

8. Conclusion

It is our contention that by dividing the task of establishing and analyzing a large scale economic data base into two subtasks, the designer will be able to develop a more efficient system requiring the investment of fewer resources. The first subtask would be the development or acquisition of a generalized data management system, which will support complex data structures, and will provide the user with a query capability. The second subtask would be to develop the economic analysis and data reduction programs using the data management system as the foundation. This general framework will also provide a strong foundation for studies to be carried out in the future.

Carnegie-Mellon University
Purdue University

Bibliography

[1] David, M. H., W. A. Gates and R. F. Miller, *Linkage and Retrieval of Microeconomic Data,* Lexington Books, 1974.

[2] Rausser, G. C. and R. Howitt, "Stochastic Control of Environmental Externalities", *Annals of Economic and Social Measurements,* Spring, 1975.

[3] Haseman, W. D., A. Z. Lieberman, J. F. Nunamaker, and A. B. Whinston, "*Generalized Planning System/Data Management System (GPLAN/DMS), Users Manual,*" Krannert Graduate School of Industrial Administration, December, 1973.

[4] Haseman, W. D., J. F. Nunamaker, and A. B. Whinston, "The CODASYL DBTG Report as an Extension to FORTRAN", *Management Datamatics,* October, 1975.

[5] CODASYL Committee, *The CODASYL DBTG Report,* 1972.

[6] Cash, J. I., R. H. Bonczek, W. D. Haseman, C. W. Holsapple, and A. B. Whinston, "*Generalized Planning System/Query System (QS), Users Manual*", Krannert Graduate School of industrial Administration.

[7] Bonczek, R. H., W. D. Haseman, and A. B. Whinston, "Structure of a Query Language for a Network Database", July, 1975.

[8] Haseman, W. D. and A. B. Whinston, "Problems Solving in Data Management", *Proceedings of Fourth International Joint Conference on Artificial Intelligence,* Tbilisi, Georgia, USSR, September, 1975.

[9] Alter, H. E., "Creation of a Systematic Data Set by Linking Records of the Canadian Survey of Consumer Finances with the Family Expenditures Survey 1970", *Annals of Economic and Social Measurement,* Vol. 312, 1974.

[10] Marsden, J. R., D. E. Pingry, and A. B. Whinston, "Large Scale Data Analysis—The Identification of Outliers", Krannert Graduate School of Industrial Administration, June, 1974.

[11] Ruggles, N. and R., "A Strategy for Merging and Matching Micro-data Sets", *Annals of Economic and Social Measurement,* Vol. 312, 1974.

received October 1974
revised September 1975