DOCUMENT RESUME

ED 099 974 EA 006 591

AUTHOR	Nunamaker, J. F., Jr.; And Others
TITLE	GPLAN: A Generalized Data Base Planning System.
INSTITUTION	Purdue Univ., Lafayette, Ind. Herman C. Krannert Graduate School of Industrial Administration.
REPORT NO	Pap-392
PUB DATE	Jan 73
NOTE	47p.
EDRS PRICE	MF-$0.75 HC-$1.85 PLUS POSTAGE
DESCRIPTORS	*Data Bases; Decision Making; *Information Systems; *Management Systems; Models; *Planning; Problem Solving; Simulation; *Systems Development
IDENTIFIERS	*Generalized Data Base Planning System; GPLAN

ABSTRACT

It is recognized that there is a major gap between the promises of large data bases and optimization and simulation models and their actual ability to solve real world problems. This document describes a Generalized Data Base Planning System (GPLAN), currently being developed at Purdue University, that is proposed as a system to bridge this gap. A number of systems contain some of the components of GPLAN, and a survey of each system is presented with details of its contribution to GPLAN design. Research on two systems at Purdue is directly leading to the development of GPLAN. One application is concerned with the development of a regional water pollution control planning system; the other is an interactive information system design package. The authors suggest that GPLAN is as much a major advance over a Generalized Data Management System (GDMS) as a GDMS is a major advance over application programs with formatted sequential files. A bibliography is included. Diagrams may reproduce poorly. (Author/DN)

GPLAN:
A GENERALIZED DATA BASE PLANNING SYSTEM

by

J. F. Nunamaker, Jr.
D. E. Swenson
and
A. B. Whinston

Paper No. 392 - January 1973

Institute for Research in the
BEHAVIORAL, ECONOMIC, and
MANAGEMENT SCIENCES

KRANNERT GRADUATE SCHOOL OF
INDUSTRIAL ADMINISTRATION

Purdue University
West Lafayette, Indiana

GPLAN: A Generalized Data Base Planning System

J. F. Nunamaker, Jr. [1,2]

D. E. Swenson [1]

A. B. Whinston [2,3]

September, 1972

Purdue University
West Lafayette, Indiana

1 Computer Sciences Department
2 School of Industrial Administration
3 Economics Department

GPLAN: A Generalized Data Base Planning System

by

J. F. Nunamaker, Jr., D. E. Swenson, and A. B. Whinston

Purdue University

ABSTRACT

It is recognized that there is a major void between the promises of large data bases and optimization and simulation models and their actual use to solve real world problems. A Generalized Data Base Planning System (GPLAN), currently being developed at Purdue University, is proposed as a system to bridge this gap. A number of systems contain some of the components of GPLAN, and a survey of each system is presented with details of its contribution to GPLAN design.

Research on two systems at Purdue University is directly leading to the development of GPLAN. One application is concerned with the development of a regional water pollution control planning system, and the other application is an interactive information system design package.

A Generalized Data Base Planning System is not just a new name for A Generalized Data Management System (GDMS), but represents a new generation of software. GPLAN is as much a major advance over a GDMS, as a GDMS is a major advance over application programs with formatted sequential files.

GPLAN: A Generalized Data Base Planning System

J. F. Nunamaker, Jr., D. E. Swenson, and A. B. Whinston

Purdue University

Introduction

Societal problems and industrial problems, if they are to be studied analytically, will require two broad components: first, a structure or model to analyze the interaction of the variables; and second, a large scale data base. The data base may be used in various ways, such as validation of the model's parameters, input data for actual runs of the model, and simply providing the data itself for other interests. While the ability to devise meaningful models with appropriate supporting data is of primary importance for the advancement of our capacity to serve societal and industrial needs, the possibility of integrating the data base handling techniques with techniques of simulation and optimization will greatly facilitate this work.

With the advent of large scale computers and complex operating systems during the last six years, the capacity exists for the processing of large scale applications. In addition, much work has been done with respect to the development of Generalized Data Management Systems (GDMS),which were designed to handle and manage large data bases. Typically, such systems have been used by management of large industrial and military organizations. These systems are used for handling inventory, receivables, customer accounting and billing, quality control and other administrative tasks.

Generalized Data Management Systems have contributed to increased use of computers, but a major void still exists. The problem of many application programs interacting automatically with a single data base remains a major barrier to general use of an information system as a planning system.

The use of an information system as a planning system will come about only when a methodology exists for automatically creating the data files needed by the many application programs and answering general queries of the data base at the same time. What is needed is a software system that, as a result of a specific query from a user, can: 1. retrieve the data necessary to answer the query; and/or 2. set up the application program or model that must be run to answer the query. This requires that the data be automatically retrieved and arranged in the proper format required by the model.

The value of such a software system is based upon: 1. the efficiency of storing and retrieving data; and 2. the range of services provided through the interactive query system.

The planning system must be designed so that the user is freed from the mundane task of data preparation, which can be tedious and frought with human errors, in order to run a model. Often, the user is not familiar with the problems and procedures of data handling, and, in most cases, would prefer not being bothered with the data handling at all.

A user who has just queried a data base will have gained very little if he must further select, re-arrange, reformat, and punch his retrieved data for input to an application program or model. Thus what is needed is a Generalized Data Base Planning System (GPLAN), which results from the extension of a Generalized Data Management System to handle the automatic setup of models from a Data Base as instructed by the user through the Query Language. GPLAN is a natural extension of GDMS and represents the next generation of GDMS's.

In many areas, there has been considerable work on development of simulation and optimization packages, with the end result that these packages are not really used by the people who should be using them. The reasons for this huge investment in application packages with little resultant usefulness are simple. The packages are so difficult to use, requiring either very much technical knowledge in the particular mathematical programming technique and/or complex file setup and manipulation steps, that few people are able to use the package without relying heavily on technical help or considerable education in the specifics of each particular package.

It is obvious that to increase the usefulness of simulation and optimization packages, the nontechnical and/or management personnel who are knowledgeable in the general area of endeavor should be able to easily use these packages.

What is proposed here is to generalize data base planning systems. For a specific area of human endeavor needing such a system, this would mean that it would be easy to set up a data base, link it with application packages, and query it in a meaningful manner -- in short, it would be easy to set up a generalized planning system. This would mean that ad hoc solutions to specific areas would be replaced with a generalized approach comprehensive enough to solve many of the problems occurring in setting up specific planning systems.

In order to take advantage of the knowledge that can be gained from the consideration of specific systems, a regional water pollution control planning system is described briefly and used for some examples throughout the rest of the paper. This system is discussed in order to develop the proper motivation for a Generalized Planning System.

Water Pollution Control Planning System

The purpose of the water pollution control planning system is to develop a plan that minimizes the cost of pollution abatement structures while satisfying a set of water quality goals throughout an entire river basin. This planning system uses the most prevalent measure of water quality in use today, the level of dissolved oxygen concentration.

The constraints of this model are constructed by dividing the river into sections and constraining the water quality, interpreted as the dissolved oxygen deficit level, to be met at the end of each section. Starting from the headwaters of the river, new sections are defined whenever the river parameters change significantly, such as effluent flow entering the river, incremental flow entering the river (tributary flow, ground water, etc.), the flow in the main channel being augmented or diverted, or the parameters describing the river changing gradually over a longer distance.

The quality constraints are sequentially dependent in that the quality in each section is a function of the quality in the preceding section. But the possibility of tributaries, flow augmentation, and incremental and effluent flows entering at downstream points, complicates the relationship between the constraints.

Three possible treatment techniques are allowed for in the model: 1. by-pass piping; 2. regional and on-site treatment plants; and 3. flow augmentation. Thus piping flows are allowed from each polluter to each river section, from each polluter to each treatment plant, and from each treatment plant to each river section.

In addition to quality constraints, flow conservation constraints are needed for both the polluters and the treatment plants.

The solution technique of the model is the use of a general purpose non-linear algorithm adapted for this model. The major problem involved in adapting the algorithm was the calculation of the partial derivatives of both the constraints and the objective function. These partials were necessary to set up a local Linear Programming problem to determine the direction of search in the stepwise nonlinear problem. Starting with a point in the domain of the objective function, a new point is calculated from it by making a step to either reduce the value of the objective function, if the original point is a feasible solution to the nonlinear programming problem, or obtain a "more feasible" solution if the point is an infeasible solution ("more feasible" by reducing the value of the most infeasible constraint).

The cost function essentially involves the costs of new or upgraded treatment plants, reservoirs for flow augmentation, and various costs of new pipes to or from the polluters, treatment plants, and river sections.

Difficulty with the Present Approach

Let us take an example of solving a mathematical programming problem (although any complex application problem would be similar), such as the nonlinear programming model discussed in the water pollution control planning system. A programmer with knowledge of mathematical programming theory, or a group of people which together has the required knowledge, would write and check out a program to solve the specific problem. Then data would be gathered and stored in the format necessary for the program. To check out the data, separate programs would have to be written to test each part of the input data file for which testing was needed. Several sets of corrections to the data file would probably need to be made.

The programmer(user) would then have to fill out some control cards giving various parameters of his data. Obtaining these parameters might involve some manual calculations and may require running some other programs on the original data.

Finally, the mathematical model could be run and, with several iterations and interpretations by the knowledgeable mathematical programming person, the problem could be solved. A report would have to be written explaining the problem and its computer solution. To solve a similar problem would take almost the same steps, except that a lot of the existing programs could probably be used, although major revisions are also possible.

The existing data on a file for solving this problem is most likely useable only for this application, even though parts of it may be useable for other applications.

The time lag between problem recognition and problem solution will be too long and the cost will be expensive. It is possible that, without countless hours of detailed documentation, the program written is unuseable except for one or a few people.

If we consider the water pollution control planning system, set up as an application program with a file, the information on treatment plant cost data and the parameters of the rivers cannot be used easily by anyone other than the developers of the system, unless the new user knows the specific format of the data and how it is stored on auxiliary memory.

Motivation for the Development of Generalized Data Base Planning Systems

We must define what a Generalized Data Management System is, before we define the characteristics of a Generalized Data Base Planning System.

Groner and Goel[3] define and characterize a Generalized Data Management System as follows:

"A GDMS consists of data, structure, and a set of algorithms for manipulating the data and the structure. It acts as a communication channel between the user community and the data base. Minker[4] characterizes a GDMS as follows: 'A data management system is considered generalized when it permits the manipulation of newly defined files and data with the existing programs and systems.' [A GDMS] facilitates reference to data by name and not by physical location,' and, '[it] facilitates the expression of logical relations among data items.' A well designed GDMS permits users to access and manipulate elements of the data base in a way that is both natural and convenient for them and efficient in terms of its system utilization.

"Much of the benefit derived from a GDMS results from the insulation of the people and programs from the data. In conventional systems the structure of the data must be explicitly embodied in each program accessing the data. This limits the application of programs to data whose structure has been defined to them. It also limits the ability to restructure data in response to new needs without modifying every program embodying the old structure. These limitations on applicability of programs and the flexibility of the data base impose a rigidity upon conventional data processing systems. While this rigidity is not serious in repetitive well defined tasks such as payroll or inventory control, it is a decided obstacle to the successful performance of systems that must respond to continually changing requirements.

"GDMS systems evolved from sequential formatted file systems. This evolution has been in the direction of more complex logical data structures and more complex operations upon them."[11]

The work on Generalized Data Management Systems until now has focused on two related, but distinctly different approaches to the design of data management systems.[5] The first approach involved the design of a special query system for retrieving data from a data base. This approach permitted new programmers to readily access data and to ask sophisticated questions of the data base. The query capability made this approach very popular, but it has a serious drawback. Namely, that it is extremely difficult to process other applications written in FORTRAN, COBOL, PL/1, etc., against the data base. Examples of the first approach are Informatics Mark IV and GIS of IBM. As a result of this deficiency, many people preferred the second approach which involved extensions to host languages (FORTRAN, COBOL, etc.) that gave the programmer some general file handling capability. However, the non-programming user found this approach undesirable, since, if he wanted a query answered, he had to write the program(s) in the host language to retrieve the data. Examples of the second approach are General Electric's IDS (Integrated Data Store) and Burrough's Disk FORTE (Disk File Organization Technique).

Now the CODASYL Data Base Task Group[6] has proposed (see Appendix A) a solution to the problem, but in its specifications has given first priority to the host language approach, with COBOL as the first language. To interface between a host programming language and a data base, a specific subschema Data Description Language and an appropriate Data Manipulation Language need to be provided. When an interface is provided, applications still must be implemented in the specific host language system. (While this may not be a problem when and if the standardized implementation of CODASYL DBTG concepts occurs throughout the industry for the major higher level languages, it is quite a drawback for several years to come.) Even if CODASYL's concepts were all implemented, we are still without a system that a planner or manager could easily use. GPLAN is proposed as such a system.

GPLAN is a synthesis of components from other systems. There are a number of systems that exhibit some, but not all, of the features of GPLAN. Examples of some of these systems are:

1. NAPSS: Numerical Analysis Problem Solving System.[7]
2. SODA: Systems Optimization and Design Algorithm.[8]
3. GDMS: Generalized Data Management Systems.[4,5,6]
   (a) System 2000 - MRI[9]
   (b) RAMIS - Mathematica, Inc.[10]
   (c) IMS - IBM[11]
   (d) DISK FORTE - Burroughs Corporation[12]
4. OPTIMA.[13]

Numerical Analysis Problem Solving System (NAPSS)

A system that is a special case of GPLAN is the Numerical Analysis Problem Solving System (NAPSS) designed and implemented at Purdue University.

A long recognized goal of Computer Science has been to facilitate the stating of problems in languages appropriate to the specific fields in which the problems exist, and then to provide for this solution without the services of highly trained programmers and analysts. These systems are "problem solving systems" and their languages are problem-oriented languages. Thus the aim of the NAPSS project is to make the computer behave as if it had some of the knowledge, ability, and insight of a professional numerical analyst.

A user unskilled in numerical analysis can describe relatively complex problems in a simple mathematical language. Then the system selects algorithms, performs analyses, and gives diagnostics of possible difficulties and meaningless results.

The problem-oriented language of NAPSS uses some of the applicable notation of Fortran, Algol, and PL-1, since they have quite similar facilities for describing computational algorithms. But it goes beyond these languages to include mathematical concepts such as integration, differentiation, algebraic and differential equations, and approximation as part of the basic language.

The basic approach to the system design is through the development of polyalgorithms which become the numerical analysis packages that are the essential elements of the problem solving system. "A polyalgorithm is formed by the synthesis of a group of numerical methods and a logical structure into an integrated procedure for solving a specific type of mathematical problem."[1] The goal of a polyalgorithm is to combine a number of algorithms (corresponding to numerical methods) with a strategy for their selection, and use a procedure which is relatively efficient and very reliable.

The NAPSS system exists as an extension of a procedure-oriented language in an environment, permitting both on-line and remote use of the system. Initially, NAPSS statements were treated as macro-statements in Fortran and were expanded in-line into a sequence of Fortran statements which were input to an incremental Fortran processor. While NAPSS operates most frequently in an online time-sharing environment, it will accept programs submitted for batch processing. Parameter values that are needed by NAPSS can be entered at a terminal in conversational mode or be present on a standard or user-named input file for either conversational or remote mode.

In figures 1a and 1b we can see how NAPSS can be put into a more general structure with a renaming of its components.

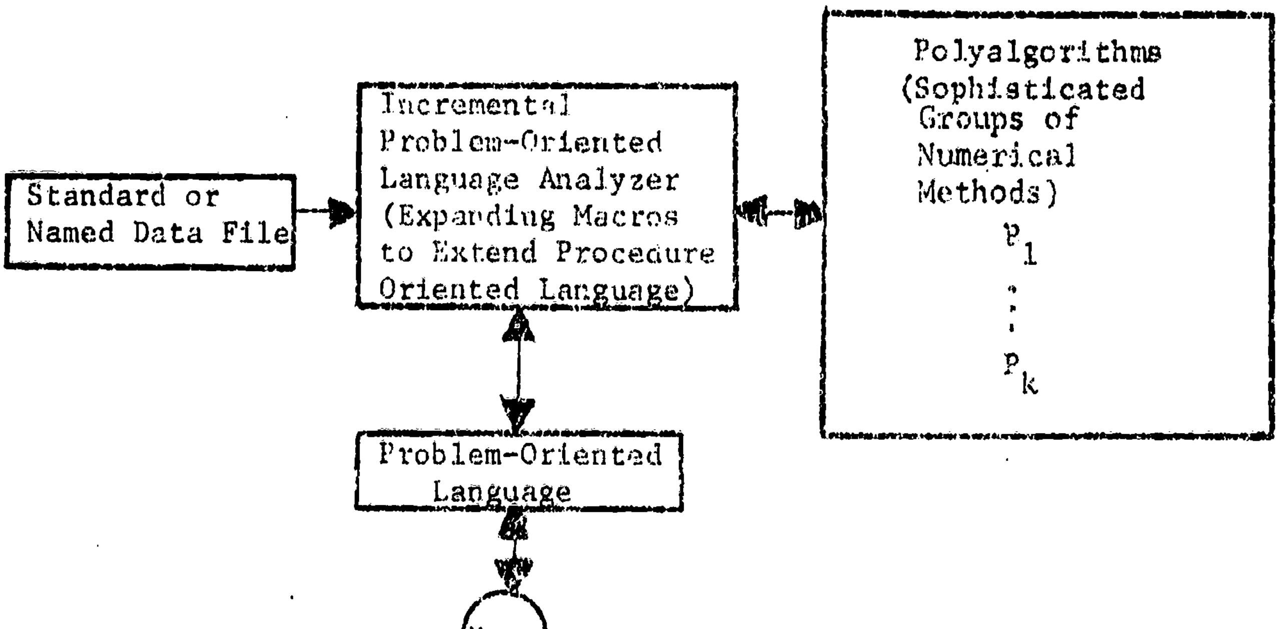
Standard or Named Data File
Incremental Problem-Oriented Language Analyzer (Expanding Macros to Extend Procedure Oriented Language)
Polyalgorithms (Sophisticated Groups of Numerical Methods)
P_1
P_k
Problem-Oriented Language
User

Figure 1a: NAPSS: Numerical Analysis Problem Solving System

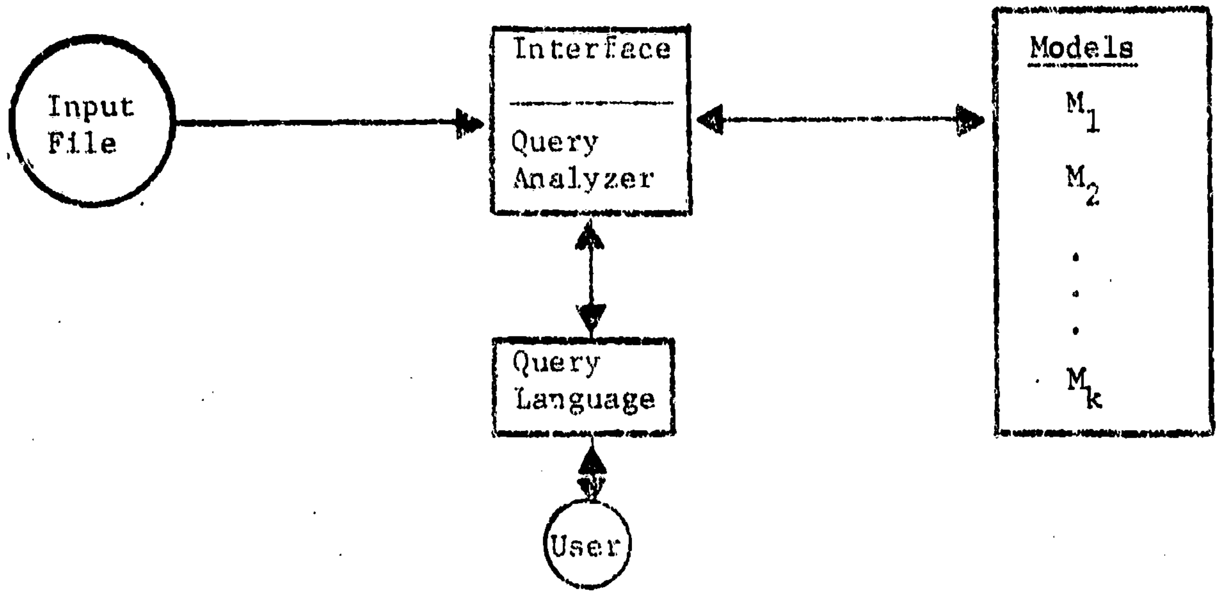

Figure 1b: NAPSS - GPLAN Terminology

SODA (Systems Optimization and Design Algorithm)

SODA is a computer-assisted decision making system for the design of information processing systems. SODA generates a complete information systems design, along with cost/performance projections of how the designed system will perform on a specified hardware/software configuration. SODA consists of four major components:

SSL:SODA STATEMENT LANGUAGE

SSA:SODA STATEMENT ANALYZER

SGA:SODA GENERATOR OF ALTERNATIVES

SPE:SODA PERFORMANCE EVALUATOR

SSA is a computer program that analyzes the requirements of an information processing system stated in SSL. The Statement Analyzer also provides feedback information to the user to assist him in achieving a better problem statement.

SSA also produces a number of networks which record the interrelationships of processes and data and passes the networks on to SGA and SPE.

Each type of input and output is specified in terms of the data involved, the transformation needed to produce output from input and stored data. Time and volume requirements are also stated. SSA analyzes the statement of the problem to determine whether the required output can be produced from the available inputs. The problem statement stored in machine readable form is processed by SSA which:

1. checks for consistency in the Problem Statement (PS) and checks syntax in accordance with SSL; i.e., verifies that the PS satisfies SSL rules and is consistent, unambiguous, and complete.

2. prepares summary analyses and error comments to aid the problem definer in correcting, modifying and extending his PS.

3. prepares data to pass the PS onto SGA, and

4. prepares a number of matrices that express the interrelationship of Processes and Data Sets.

SGA is a procedure for the selection of a Computer System(cpu, core size, auxiliary memory devices) and the specification of alternative designs of program structure and file structure. SGA constructs a configuration of equipment in order to evaluate performance of the system. A number of models are used (to compute timing estimates) that select timing factors for alternative hardware/software configurations from a data file. SGA simulates the jobstream as it would be processed on the selected configuration, and, using the factors from the hardware/software library, SGA and SPE produce detailed cost/performance projection reports so that the user can evaluate the final design.

There are a number of systems similar to some aspects of SODA, such as SCERT[14,15] and CASE.[16]

In Figures 2a and 2b it is shown how SODA fits into a more general structure.

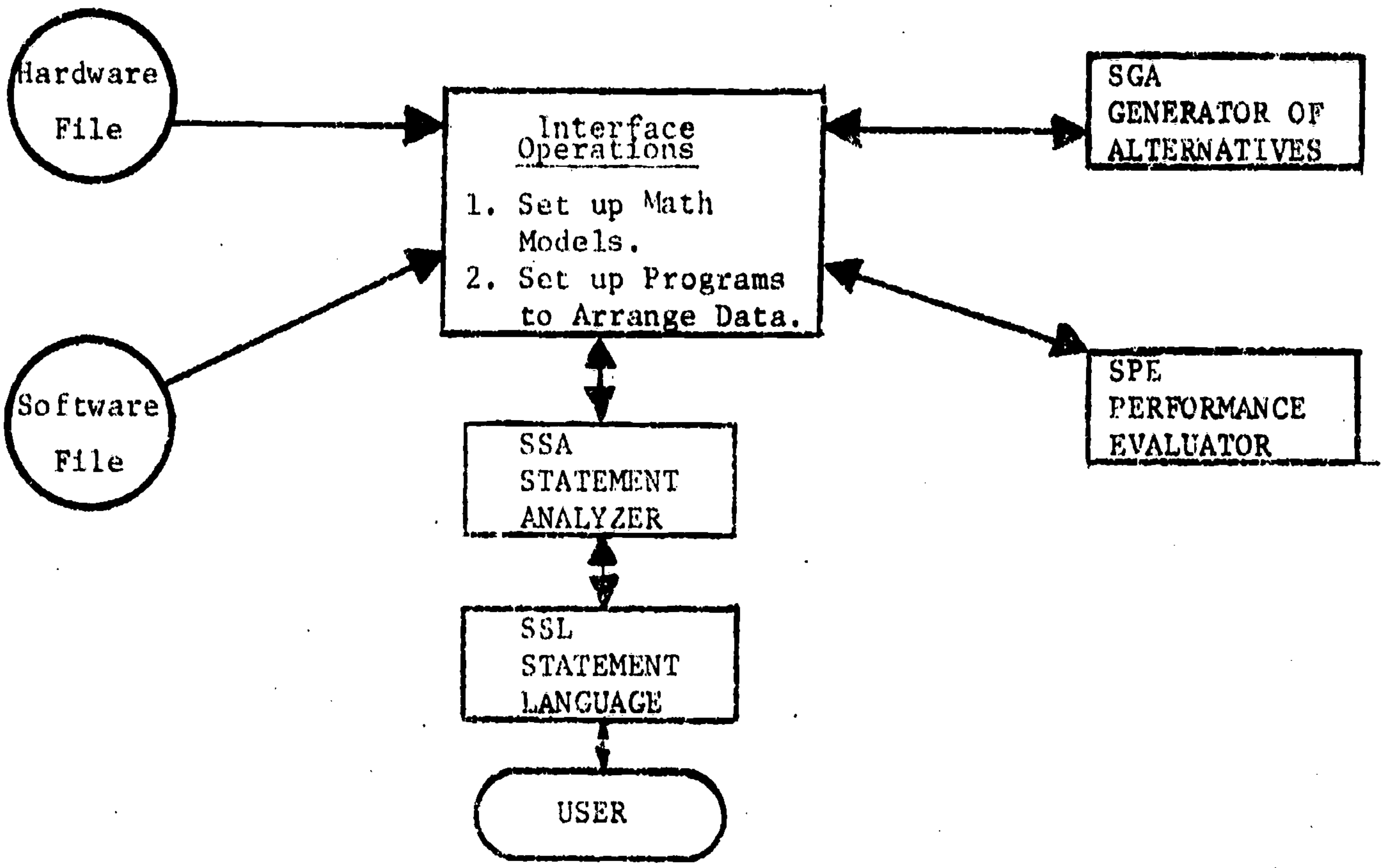

Figure 2a: SODA: Systems Optimization and Design Algorithm

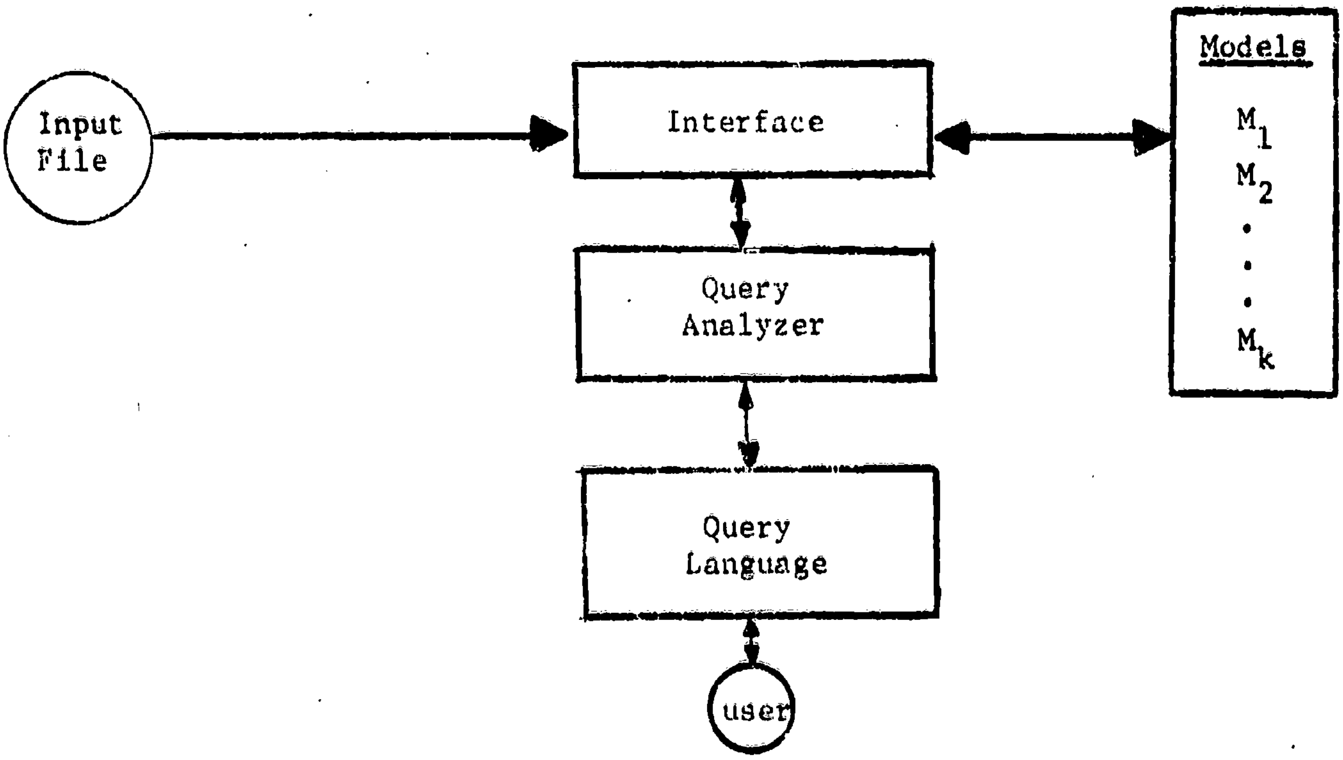

Figure 2b: SODA: GPLAN Terminology

Generalized Data Management Systems

SYSTEM 2000

SYSTEM 2000, developed by MRI Systems Corporation, is a general-purpose data base management system. The basic system provides a comprehensive set of data base management capabilities, including the ability to define new data bases, modify the definition of existing data bases, and retrieve and update values in these data bases.

In SYSTEM 2000, the basic components of data base definitions are data elements and repeating groups. Values are stored in data elements. Repeating groups describe a structure for storing multiple sets of data values (data sets) and also serve to link hierarchical levels of the definition.

Values for each eleme t and logical entry (record) may vary in length. The user may specify without restriction which elements in the data base are to be inverted and become key fields, and what hierarchical relationship an element will have with other elements in the data base. Data security is maintained by password control to the data base and additional password control to each component.

The Procedural Language feature of SYSTEM 2000 enables users to manipulate data in a SYSTEM 2000 data base from COBOL or FORTRAN. This feature provides the mechanism to address any part of the data base of interest to the procedural program, to retrieve data in a sequence and format suitable for procedural processing, and to update the data base from the program.

RAMIS: Random Access Management Information System

RAMIS, developed by Mathematica, Inc., is a data base management system which permits a user to describe and build data bases, maintain the data in the data bases through updates, additions, and deletions, retrieve information from the data bases and display it in meaningful report formats, or pass the information to other processing programs.[4]

RAMIS is both a report generator and a data management system, since it has a simple and logical English-like language, which permits the user to both request information from data bases, and, at the same time, process it into finished reports.

RAMIS organizes the physical placement of data into tree structures on random access devices by exploiting the hierarchical relationships of the data fields. The user has to supply only some minimal information about these relationships. User written programs in Fortran, Cobol, Assembler, or PL/1 can also be linked directly into RAMIS.

IMS: Information Management System

The Information Management System (IMS) is a system designed to facilitate the implementation of medium to large common data bases in a multi-application environment.[5] This environment is created to accomodate both online message processing and conventional batch processing, either separately or concurrently. The system permits the evolutionary expansion of data processing applications from a batch-only to a teleprocessing environment.

The data base processing capabilities of IMS are provided by a facility called Data Language/I. The data base functions supported are definition, creation, access, and maintenance. The full data base capabilities of Data Language/I can be used in the IMS batch processing or teleprocessing environment.

Data communication capabilities are characterized by the use of input/output terminals in remote and local environments, connected to the computer, which provide the user with access to the data base. IMS also has extensive message scheduling, checkpoint, and restart facilities.

DISK FORTE

Disk FORTE, the Burroughs manufacturer system, is programmer-oriented at the most basic level. Nearly all features and capabilities of other data management systems must be programmed in Disk FORTE. Yet, it permits both hierarchic and network data structures (user-programmed, of course), which make possible more complex associations among data.

Disk FORTE makes its data management capabilities available through extensions to COBOL which are handled by a pre-compiler.

Figure 3 shows the generalized structure of SYSTEM 2000, RAMIS, IMS and DISK FORTE.

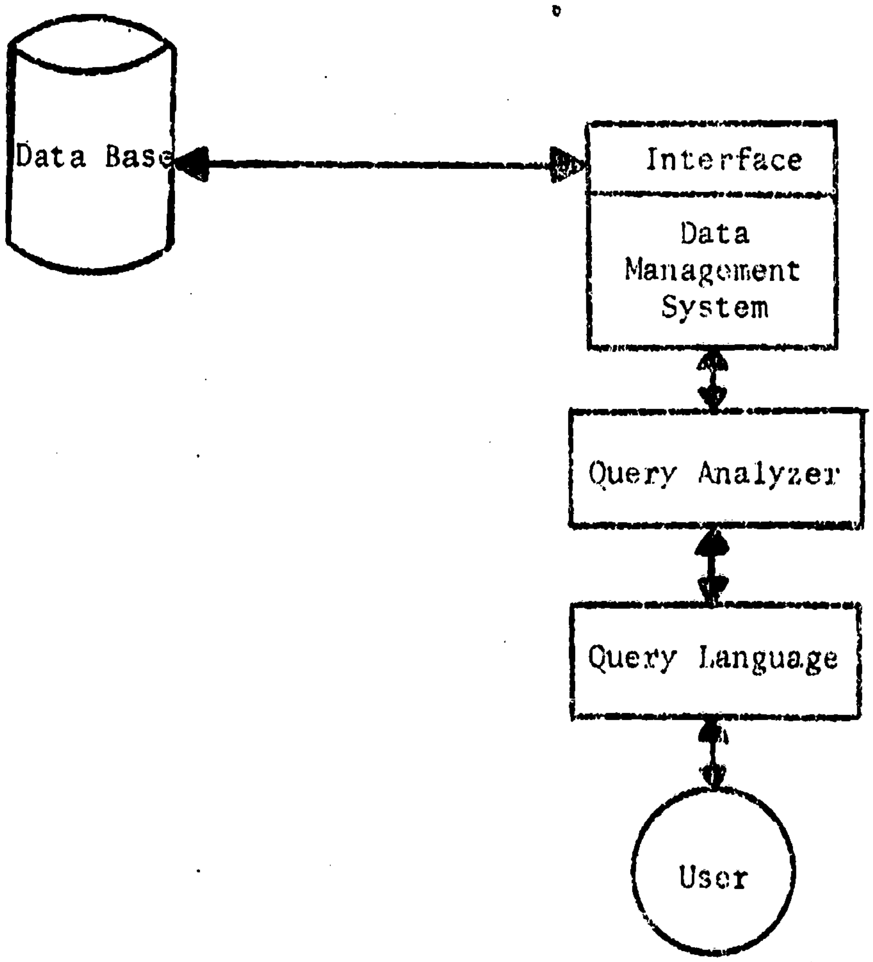

Figure 3: Generalized Data Managemant System

OPTIMA

OPTIMA is an advanced mathematical programming system for the CDC 6000 series computers. It includes advanced algorithms and techniques in addition to algorithms for standard linear programming formulations. User-controlled data and storage management features are also provided.

"The basis of OPTIMA is a revised product-form, composite, bounded variable, separable, multipricing, simplex, linear programming algorithm." Some of the advanced features that OPTIMA provides are: the capability to form a nontrivial starting basis; the ability to start a solution using a previous basis; dynamic control of the frequency of inversion of the basis matrix; provision for partial and multiple pricing; the use of maps to exclude or include specified vectors in the basis; and elaborate recovery procedures. A dual optimization algorithm is available for those problems in which its use might be advantageous; and postoptimal analysis of a problem can be accomplished as an integral part of OPTIMA.

Through the use of the Applications Control Language (ACL), OPTIMA allows dynamic control of the progress and execution of the program. ACL has logic and computational capability and provides verbs and phrases for modifying various parameters and controlling the progress of the solution.

An ACL program must be written for any study. This program defines the data files to be used and the operations to be performed on these files, sets any parameters and controls necessary, and calls various routines required to carry out the study.

Two other languages, the Matrix Generator Language (MGL) and the Report Generator Language (RGL) operate within control of the ACL. MGL provides capabilities for generating a problem matrix automatically. RGL provides the capability for generating reports in any desired format, and permits computer-generated solutions to be used for further arithmetic and logical computation.

Summary of Comparisons

Figure 4 shows the structures of GPLAN. In considering the four information processing systems 1) NAPSS, 2) SODA, 3) GDMS, and 4) OPTIMA, it is observed that we must have at least a query language, a query analyzer and a Generalized Data Management System. NAPSS and SODA are missing the data management capabilities; the GDMS's are missing those components needed to readily interface models; and OPTIMA is missing the data management and query capabilities. These differences are elaborated on in Table 1.

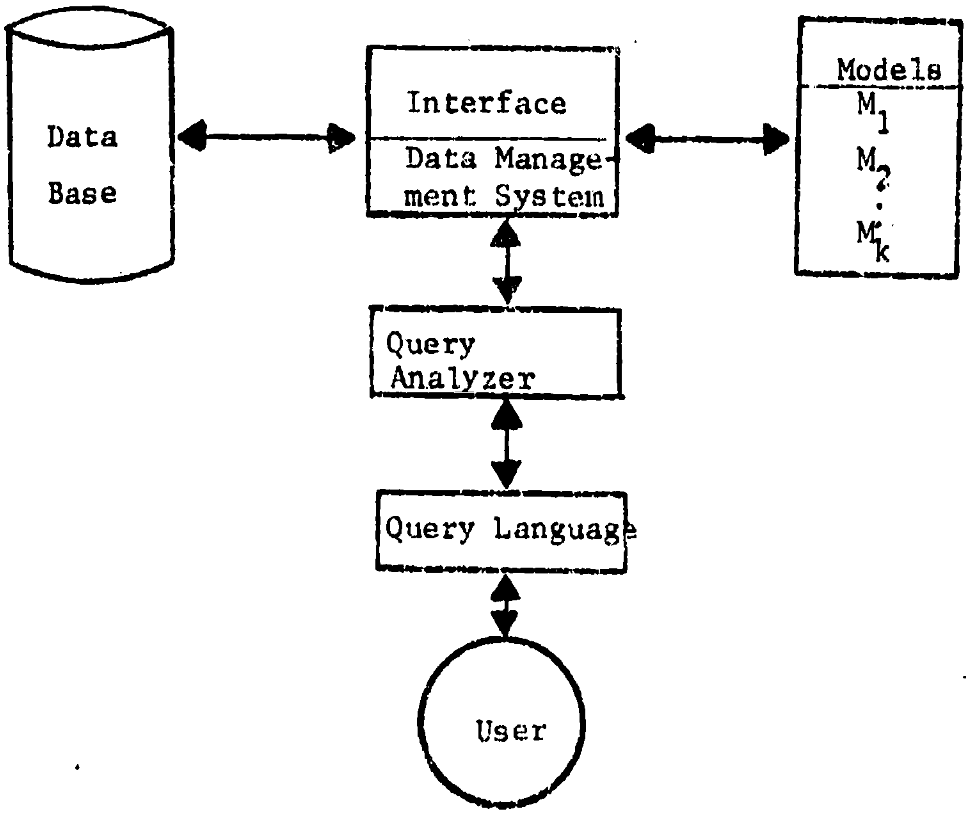

Figure 4: GPLAN: Generalized Data Base Planning System

Characteristics	DISK FORTE	GPLAN	IMS	SODA	NAPSS	OPTIMA	RAMIS	SYSTEM 2000
Mnemonic value of name	Disk File Organization Techniques	Generalized Data Base Planning System	Information Management System	Systems Optimization Design Algorithm	Numerical Analysis Problem Solving System	.....	Rapid Access Management Information System	....
Implementers	Burroughs		IBM	Purdue U., University of Michigan	Purdue University	Control Data Corporation	Mathematica, Inc.	MRI Systems Corporation
Hardware	B2500, B3500	CDC6500 IBM370	IBM/360, 370	UNIVAC 1108 IBM 360/67 CDC 6500	CDC6500	CDC6000	IBM/360,370	IBM/360,370 UNIVAC 1106,1108 CDC 6000, Cyber 70
What is the system?	Manufacturer's DBMS		Manufacturer's DBMS	Information System Design	Numerical Analysis System	Mathematical Programming System	Nonmanufacturer's DBMS	Nonmanufacturer's DBMS
System Implementation language	COBOL	FORTRAN BAL/COMPASS	BAL	FORTRAN	FORTRAN		FORTRAN and BAL	FORTRAN and BAL
User Language	COBOL extensions	OWN	COBOL, PL/1, BAL	OWN	OWN	OWN	OWN	Assembler COBOL, FORTRAN
File Organization (Storage Structures - under user control if visible to user)	Unordered Random Index-Random, Index-Sequential (visible to user)	Special File Structure	Sequential Index-Sequential Direct access (visible to user)	Sequential	Sequential	Sequential	Special File Structure	Special File Structure
Data Structures Allowed	Hierarchic or Network	Hierarchic or Network	Hierarchic	Sequential	Sequential	Sequential	Trees	Hierarchic
Storage Media	Head-per-track Direct Access Devices	Direct Access Devices	Direct Access Devices	Direct Access Devices	System Files	System Files	Direct Access Devices	Direct Access Devices
Users:								
Nonprogramming		X		X			X	X
Programming	X	X	X		X	X		X
User Language Classifications:								
Tabular	X			X				
Command			X	X	X	X		X
Free		X					X	
Procedural	X		X			X		X
Nonprocedural		X			X		X	

Characteristics	FORTE	GPLAN	IMS	SODA	NAPSS	OPTIMA	RAMIS	SYSTEM 2000
Query Language:								
Data Base Queries Allowed?	(user programmed) X	X	X				X	X
Application Packages or Models Queries Allowed		X		X	X	X		
Problem Oriented Language		X		X	X	X		
Query Analyzer:								
Basic Data Base Queries	(user programmed) X	X	X				X	X
Application Package or Model Queries		X		X	X	X		
Problem Oriented Language		X		X	X	X		
Expandable Automatically		X						
Application Packages or Models:								
Mathematical and/or optimization models		X		X	X	X		
Statistical Packages		X						
Simulations		X		X				
Any Program	COBOL (user programmed)	X						
Any package (group of interconnected programs)		X						
Comparison with GDBPS Components:								
User	X	X	X	X	X	X	X	X
Query Language		X	X	X	X	X	X	X
Query Analyzer		X	X	X	X	X	X	X
Data Management System	X	X	X				X	X
Interface	X	X	X	X	X	X	X	X
Data base	X	X	X	X			X	X
Models		X		X	X	X		

Table 1 Comparison of Information Processing Systems

Components of a Generalized Data Base Planning System

A synthesis of the previous systems results in the definition of the following nine components of a Generalized Data Base Planning System:

1. A Generalized Data Management System (GDMS)
2. Raw Data for the Data Base
3. A Query Language
4. A Query Analyzer
5. A Collection of Application Packages or Models
6. Administrative Report Module
7. User's Interface
8. Extraction Files
9. Users

Each of these components is discussed in more detail in the following sections. An overview of GPLAN for Water Pollution Control is shown in Figure 5 and Figure 6.

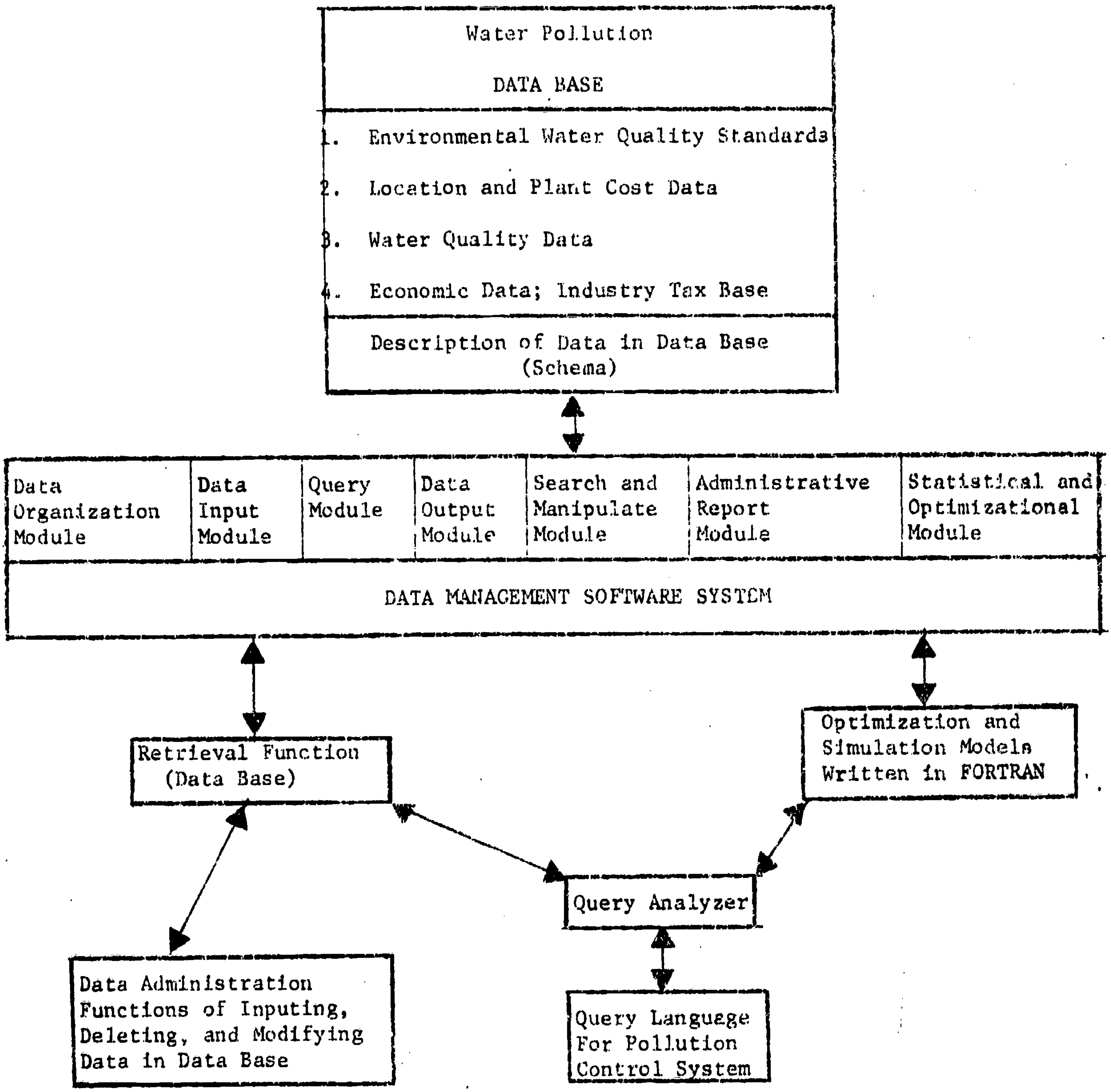
Water Pollution
DATA BASE
1. Environmental Water Quality Standards
2. Location and Plant Cost Data
3. Water Quality Data
4. Economic Data; Industry Tax Base
Description of Data in Data Base (Schema)
Data Organization Module
Data Input Module
Query Module
Data Output Module
Search and Manipulate Module
Administrative Report Module
Statistical and Optimizational Module
DATA MANAGEMENT SOFTWARE SYSTEM
Retrieval Function (Data Base)
Optimization and Simulation Models Written in FORTRAN
Query Analyzer
Data Administration Functions of Inputing, Deleting, and Modifying Data in Data Base
Query Language For Pollution Control System

FIGURE 5: GPLAN for Water Pollution Control

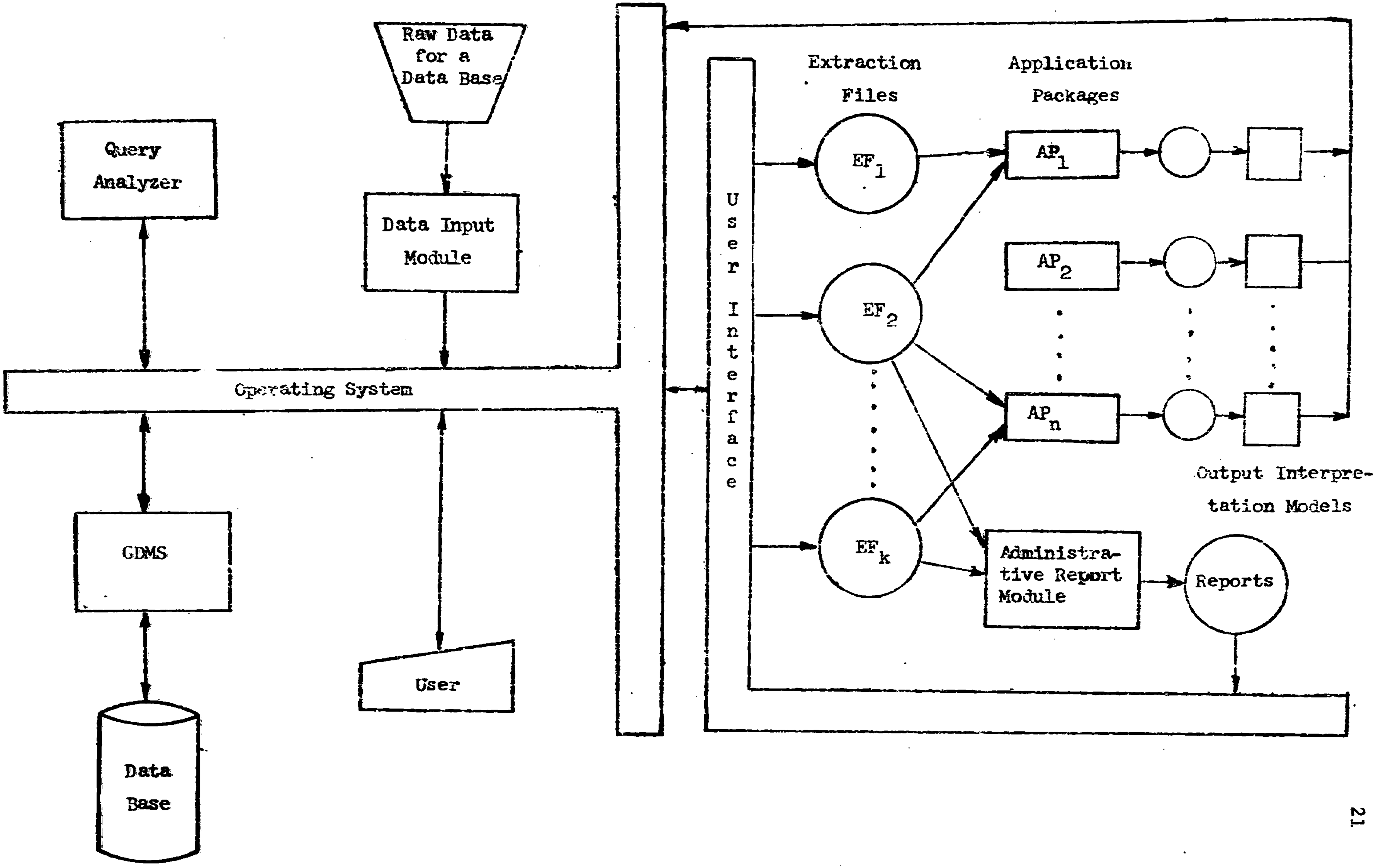

Figure 6: GPLAN: Generalized Data Base Planning System

1. A Generalized Data Management System (GDMS)

A Generalized Data Base Management System that is implemented at a particular installation under a specified operating system must be available. The GDMS must meet minimum requirements as to data structure definition, data base loading, data base updating, and data base retrieval, and it must satisfy some minimum set of queries, as specified in the following section.

Six general functions must be provided by the GDMS:

Input - the system accepts data values or information about data structures.

Search - the system searches the data base by examining the descriptions of data structures and storage structures to ascertain the existence and location of certain data values.

Storage - the system accesses a data base to add, insert, modify, or delete data values.

Maintenance - the system generates or modifies descriptions of data, data structures, and storage structures to adapt to change.

Retrieval - the system accesses a data base to obtain data values previously stored.

Output - the system exhibits data values or information about data structures and storage structures.

2. Raw Data for a Data Base

The raw data for the planning system must be available in whatever form it can be collected. A Data Input Module is used to convert the raw data into a form necessary to be loaded by the GDMS into the data base.

To refer to the data in the data base, the following terms, adapted from the CODASYL Data Base Task Group Report, are defined:

DDL - Data Description Language. A language for defining data and their relationships. The DDL is divided into schema and sub-schema.

Schema - That part of the DDL which defines the "universal" data base.

Sub-schema - That part of the DDL which describes the data known to each application program or model.

A schema describing the data on the data base must be prepared. (Note that the DDL is not necessarily the one defined in the DBTG report.)

While preliminary Data Input Modules would be dependent on the format of the raw data and dependent on the GDMS, it is hoped that some measure of independence can be achieved in the same manner as the data transformations used in implementing the user interface mentioned later.

3. A Query Language

A Basic Query Language (BQL) defines all those queries that can be handled by the GDMS alone and allows the user to request that the DBMS display certain data or answer questions about the data. The BQL is a comparative and computationally oriented language used to compare item values with other item values, or with constants, or with results of computations. Arithmetic operators define the computations to be performed, and logical operators combine simple expressions into compound expressions. The BQL can be extended as application packages are added to include questions that are answered by the new application packages, i.e., each package adds a set of new query components to the BQL. The BQL, together with all the query components from the applications packages, makes up the Query Language (QL).

General capabilities provided by the QL are:

Selective retrieval - the user specifies the selection conditions to be satisfied in retrieving the desired data.

Nonselective retrieval - the user specifies unconditional retrieval of data.

Conditional retrieval - the user employs verbs such as IF.....ELSE to test items for some qualifying values in determining alternative courses of action.

Statistical retrieval - the user may query the system about data. Statistical computations for all the instances of one item , for example, would include maximum value, minimum value, mean value, median value, mode value, standard deviation, and total number of instances.

The QL thus should provide the capability for easily asking questions of the data base and asking questions that can be handled by the various application packages. By including optimization models in the application packages, the policymaker is able to move efficiently beyond the "What if" to the "What's best" question. Moreover, much additional research needs to be done on the query language (and its associated analyzer), since its enhancement adds much to GPLAN.

As an example of the power needed in the query language, consider the added components of the planning model from the Regional Water Pollution Control Planning System. It is able to handle two distinct types of planning problems. First, it is able to select a least-cost combination of treatment methods, given water quality goals and economical, political, and water quality information from the river basin. The second type of question which can be handled is directed towards individual projects. Types of individual planning problems that could be analyzed are:

a. What is the least cost solution for towns X and Y to handle their effluent? Should they combine to construct and operate a joint treatment plant?

b. What would be the least cost solution if consideration is given to the political constraints that may become operative?

c. What is the optimal plan for capacity expansion giving consideration to the growth and shift of population and industrial growth in the basin?

d. What are the least cost and optimal treatment plans that correspond to the task of providing water of high enough quality for certain recreation activities?

e. What is the sensitivity of the optimal pollution control plan to costs and constraints?

f. What is the optimal tradoff between water quality, flow and alternative costs? For example, what would the difference in costs be if a plan was to permit the violation of a water quality standard once in 25 years as compared to once in 50 years?

GPLAN is a methodology for obtaining answers and responses to the above type of questions.

4. A Query Analyzer (QA)

A Query Language Analyzer must be able to analyze the BQL and as many application package query components as are available. A user enters his query in the QL, and the Query Analyzer analyzes the question and provides the user diagnostics to help him reformulate his question, if necessary. The query stored in machine readable form is processed by the QA which:

1. Checks for consistency in the Query and checks syntax in accordance with the Query Language; i.e., verifies the QL rules and is consistent, unambiguous, and complete.

2. Prepares error comments to aid the user in correcting, modifying and extending his Query.

3. Decides whether to pass the Query to the GDMS or one of the application packages.

4. May request additional information from the user if the action to be initiated requires it.

5. A Collection of Application Packages or Models

Application packages are simulation and optimization models, statistical packages, and other self-contained systems currently functioning under a specific computer and operating system.

We want to make it as easy as possible to add application packages, so we generalize the process by describing how we add a specific package.

We will make several assumptions about application packages: First, they are already running as batch jobs rather than interactive jobs. They may have quite long running times and making them interactive may simply mean waiting at the terminal; Second, they require user preparation to get the data ready; and third, they require other programs to run before the input data is complete.

We must know certain characteristics of an application package or model before we can consider tying it into GPLAN:

A. Input

1. For each data input, we must know what kind it is (pure data or commands) and the associated types and formats.

2. For each data input, we must know what kind of device it is assigned to (sequential or random-access).

3. We must know the passes and correct logic steps and transformations to go from the data base to each input.

4. For each data input, what must be included in system queries to the GDMS for 3 above?

B. Output

1. Is the output self-explanatory or does it require minor explanation in the form of good documentation in the Query Language description?

2. Does the output require much technical knowhow to interpret the results? If yes, an output interpretation module is required (e.g. nonlinear river basin model solution.)

C. What query components can it add to the Basic Query Language?

D. Minimal Documentation to be used by:

1. Systems personnel

2. Non-programmer researcher

6. Administrative Report Module

The Administrative Report Module will produce standard reports that must be completed and filed on a routine basis. These reports may be automatically triggered by queries or specifically requested from the console.

The standard reports will be supplied with data from the extraction file structure.

7. User's Interface

Each interface component required for each part of an applications package must be defined:

A. Simple input linkage -- direct to the Data Base.

B. Phased inputs -- including self-started and analyzed Data Base retrievals.

C. Any combination of A. and B.

D. Simple output -- direct from package.

E. Output interpretation module needed.

8. Extraction Files

Between the data base and the set of application packages and the Administrative Report Module is a set of extraction files containing those items from the data base that are used for the packages and reports. Questions to be answered on extraction files are:

A. How many should there be?

B. What items should they contain?

C. What should their data structure and storage structure be?

D. How often should they be updated?

E. If a new application package or report is added, what changes should be made in the extraction file structure?

F. Should some application packages bypass extraction files entirely?

9. Users

There are two types of users connected with GPLAN: the technical systems personnel and the nontechnical administrator or manager.

A data administrator and his systems staff are responsible for: all original data input; updating of data; restructuring the Data Base and extraction files as necessary; changing machines and operating systems; adding new packages, standard reports (Administrative Report Module), and other additions or improvements. These systems users, taken as a group, must understand fairly well every component of GPLAN. They possibly could get by with not being familiar with some of the application packages, but then would have to get consultation to patch up or improve on these.

The major group of users are non-programming administrators. These are users who don't know how to program, probably don't want to learn, and definitely shouldn't have to learn. They have a good understanding of the area for which the planning system was designed,or will have to have some training in this area before using GPLAN. Most of the details of the GPLAN implementation should be transparent to these users,and they should not notice any changes in the system,except the addition of new capabilities (possibly requested by them), new efficiency, or new packages. The success or failure of GPLAN depends on how well these users are able to carry out their querying of the data base and interaction with the application packages using only the query language and its documentation.

RFMS and RAMIS easily meet and/or exceed the minimal requirements for a GDMS as specified above. Thus all software being developed for GPLAN is being implemented on the CDC 6500 and the IBM 370/155.

Research is proceeding on the Query Language - Query Analyzer components in three areas. One area of investigation is the relationship between the QL and QA and the SODA Statement Language and SODA Statement Analyzer as used in the SODA project. Second, research on QL and QA is proceeding as a result of the development of the water pollution control models. Finally, the state of the art in artificial intelligence is being investigated for incorporation into the query language and analyzer.

Status of GPLAN

There are two major efforts under way with respect to the development of GPLAN:

1. Development and construction of the software for GPLAN.
2. Work on a real world planning system (Water Pollution Control) and development of user training aids.

1. Development of Software for GPLAN

Two different GDMS systems are being evaluated in parallel with the construction of GPLAN. Development is proceeding using RAMIS and RFMS (Remote File Management System). RFMS is a version of SYSTEM 2000 that was originally developed at the University of Texas at Austin. RFMS has been converted to run under the Purdue MACE Operating System,and substantial improvements have been added to the original version.

The two most difficult areas in the development of software for GPLAN are the User's Interface and the Extraction Files. Thus, a Data Description Language schema for data base description, and the Data Description Language subschema for the description of application package and administrative report data requirements, have been defined. Research is proceeding on the automatic mapping between the data base schema and an application package's subschema. Included in this mapping is a set of extraction files to be composed of subsets of items from the data base. An integer programming model has been defined which relates the data items of the data base to those on the extraction files as required by the application programs. Also, a cost function representing the cost of operating GPLAN has been defined. An important aspect of the problem is the optimization of the extraction files by solving for the extraction file arrangement which minimizes operating costs.

2. Work on a Real Application

Data is presently available to us from a previous study on the West Fork of the White River in Indiana for the development of a demonstration project concerning the Water Pollution Data Base Planning System. The insight achieved through the development of a specific planning system has already proved to be a tremendous aid in the accomplishment of the major goal of having a truly easy-to-use system.

The query system is being implemented in two modes:

1. Standard 80 column teletype
2. Graphics Terminal

The usefulness of GPLAN is enhanced considerably through the effective use of a graphical display. The graphics terminal offers the obvious advantage of being able to output designs, graphs, etc., in a more visable and appealing form. But the main advantage of interactive graphics is that it offers the user the capability of complete user interaction with the planning system.

Consider, for example, the river basin planning system. The optimization models output a diagram of the optimal solution to a specific water pollution control problem, showing the actual location of treatment plants and cooling towers on a computerized representation of the river basin, i.e., the actual solution is illustrated on a map of the river basin. The user may not have much confidence in the results, but he can at least relate to the output in this form. However, if he is given the opportunity to improve the solution by making adjustments to the design, or by changing the location or capacity of a treatment plant through the graphical query system, he finds that he is part of the decision making or planning process. Now, we can let the user input his own design and then compare the value of the objective function for his design with the optimal design. GPLAN can then indicate whether or not his design is even feasible. The user can also be given the opportunity to experiment with the values of the constraints and try different water quality goals. The result of this interaction is that the user has increased confidence in the planning system with a unique appreciation of the special talents and capabilities of man and machine.

The user can only be convinced that the mathematical solution is "good" if he can't improve on it himself. This experience was also supported by observations from a project concerned with the location of a major highway in southern California.[17]

Interactive graphics allows the user to utilize insight that often can't be built into models. This man-machine interaction enhances the planning system and brings the user into the decision making process.

REFERENCES

1. D. Pingry and A. Whinston, "A Survey of Planning Techniques for Pollution Control," Krannert School of Industrial Administration, Purdue University, August 1972.

2. D. Pingry and A. Whinston, "A Multi-Goal Water Quality Planning Model," Krannert School of Industrial Administration, Purdue University, August 1972.

3. Leo Groner and Amrit Goel, "Generalized Data Management Systems for Structured Information Retrieval," Working Paper, Department of Systems and Information Science, Syracuse University, August 1972.

4. Jack Minker, "General Data Management Systems - Some Perspectives," Computer Science Center, University of Maryland, December 1969.

5. CODASYL Committee, Feature Analysis of Generalized Data Base Management Systems, Association for Computing Machinery, May 1971.

6. CODASYL Committee, Data Base Task Group Report, Association for Computing Machinery, April 1971.

7. J. R. Rice and S. Rosen, "NAPSS: A Numerical Analysis Problem Solving System," Proceedings of the ACM National Conference, 1966.

8. J. F. Nunamaker, Jr., "A Methodology for the Design and Optimization of Information Processing Systems," AFIPS Proceedings, Volume 38, AFIPS Press, May 1971.

9. MRI Systems, "System 2000 General Information Manual", 1972.

10. "RAMIS Information Bulletin - Description and Specifications," Mathematica, Inc., Princeton, New Jersey, 1972.

11. IBM, "Information Management System/360, Version 2 General Information Manual," GH 20-0765-1, Second Edition, February 1971.

12. Burroughs Corporation,"DISK FORTE Manual," 1969.

13. Control Data Corporation, 'OPTIMA Version 3.0 Reference Manual," Publication Number 60207000, Revision B, 1969.

14. D. J. Herman and F. C. Ihrer, "The Use of a Computer to Evaluate Computers," AFIPS Proceedings, Volume 25, May 1964.

15. L. R. Huesman and R. P. Goldberg, 'Evaluating Computer Systems through Simulation,"Computer Journal, August 1967.

16. Computer Learning and Systems Corporation, '"The Case for CASE: Computer Aided System Evaluation," 1971.

17. Robin Liggett, Computer Graphics Staff, U.C.L.A. School of Architecture. Private Communication, July 1972.

Appendix A -- The Data Base Task Group Proposal

from: "The Debate on Data Base Management," EDP Analyzer, Vol. 10, No. 3 (March, 1972), pp. 4-5.

What was proposed

In designing the specifications for two data description languages and a data manipulation language, the DBTC was faced with many options. They could choose to embed these languages in one or more "host" programming languages, or they could choose to make them self-contained. They could orient these languages toward the application programmer, or toward the non-programming user; that is, they could be procedural or non-procedural. Or they could choose to support both the application programmers and the non-programming users.

The DBTC selected what they considered the highest priority tasks to work on. They chose the host language approach, with Cobol being one host language. And they chose to specify a DML at the same level of procedurality as Cobol. At the same time, they designed the DDL and DML so that the general data base could be interfaced with other programming languages. They deferred work on non-procedural languages as being less urgent than the procedural languages -- but they recognized the need for the non-procedural languages.

We will briefly cover some of the key terms involved. The name schema DDL was given to the source language for defining the complete data base. It was to be totally independent of any one programming language but able to interface with a variety of them through appropriate sub-schema DDL's. The term sub-schema DDL applied to those DDL entries that describe the part of the data base known to one or more specific programs, written in one programming language. A translation may be needed between the way data is stored in the data base and the form in which it is needed by an application program. This translation would be defined by the match between the schema DDL and the sub-schema DDL. An area is a logical container which can hold records and which can be mapped onto storage media. A set of records is a group of related records, associated by pointers or pointer arrays. Each set type has one record type declared as the owner, and one or more types declared as members. A data base-key is a unique record identifier, defined by the implementor, from which the area and location within the area can be determined. The implementor may choose to make the physical record address directly derivable from the database-key.

Note that a sub-schema DDL and an appropriate DML provide the interface between a host programming language and the data base. Note, too, that the DBTG proposed specifications for the Cobol sub-schema DDL and the Cobol DML.

Cobol sub-schema DDL. Four main sections of the Cobol DDL include the Renaming Section, Area Section, Set Section, and Record Section. The Renaming Section allows the data administrator and application programmer to relate names in the Cobol sub-schema to the names in the complete data base schema, to provide conformity with what the host language requires. (For example, Cobol can have 30-character names, while Fortran can have 6-character names). The Area Section allows enumerating the areas of the schema which are included in the sub-schema--and by implication, to remove from view all other areas of the schema. It is convenient, but not strictly accurate, to think of areas in terms of storage media--tracks or cylinders of mass storage, magnetic tapes, etc. So in the Area Section, those portions of mass storage which the program(s) can access are named. The Set Section allows enumerating and defining the sets of the schema that are to be included in the sub-schema and again, removing from view all other sets. Finally, the Record Section allows the naming and definition of record types within a data base.

While somewhat similar to the record descriptions in Cobol, the Record Section does differ from it in important respects. The Location Mode, copied from the schema, defines the means of accessing the record--by calculation (randomizing), direct (by database key), or by searching through a named set. The area to which a record is assigned is named. A record privacy lock can be included, for various types of access. Each data field is defined by Picture, Usage, Sign, Occurs, and Privacy Lock.

Cobol DML. Fifteen verbs are defined, for storing, retrieving, etc.; most of these have two or more optional formats, as in the case of the Cobol verbs. The verbs and the number of optional formats for each are: Open(2) Close (2), Insert (2), Remove (2), Modify(2), Order (2), Delete (1), Find(7), Get (2), Store (1), Free (1), Keep (1), Move (2), If(2), and Use (1).

The following is a listing of Institute Papers which are still in supply. Copies may be obtained from the Secretary of the Institute Paper and Reprint Series, Krannert Graduate School of Industrial Administration, Purdue University, West Lafayette, Indiana 47907.

(When requesting copies, please specify paper number.)

Paper No.	Title and Author(s)
83	A CLASS OF UTILITY FUNCTIONS ADMITTING TINTI'S HOMOGENEOUS SAVING FUNCTION, Peter Jason Kalman.
101	CLASSIFICATION OF INVESTMENT SECURITIES USING MULTIPLE DISCRIMANANT ANALYSIS, Keith V. Smith.
123	A NOTE ON KONDRATIEFF CYCLES IN PREWAR JAPAN, Charles R. Keen.
138	BOREDOM VS. COGNITIVE REAPPRAISAL IN THE DEVELOPMENT OF COOPERATIVE STRATEGY, Marc Pilisuk, Paul Skolnick, Kenneth Thomas and Reuben Chapman.
144	ON IMPLICATIONS OF PRODUCTIVITY COEFFICIENTS AND EMPIRICAL RATIOS, Harry Schimmler.
147	DEPTH, CENTRALITY AND TOLERANCE IN COGNITIVE CONSISTENCY, Marc Pilisuk.
148	THE GENERAL INCONGRUITY ADAPTATION LEVEL (GIAL) HYPOTHESIS--II. INCONGRUITY MOTIVATION TO AFFECT, COGNITION, AND ACTIVATION-AROUSAL THEORY, Michael J. Driver and Siegfried Streufert.
150	PORTFOLIO REVISION, Keith V. Smith.
154	HEROES AND HOPLESSNESS IN A TOTAL INSTITUTION: ANOMIE THEORY APPLIED TO A COLLECTIVE DISTURBANCE, Robert Perrucci.
155	REGIONAL ALLOCATION OF INVESTMENT: A FURTHER ANALYSIS, Akira Takayama.
158	TWO CLASSICAL MONETARY MODELS, Cliff Lloyd.
161	THE PURCHASING POWER PARITY THEORY: IN DEFENSE OF GUSTAV CASSEL AS A MODERN THEORIST, James M. Holmes.
162	HOW CHARLIE ESTIMATES RUN-TIME, John M. Dutton and William H. Starbuck.
163	PER CAPITAL CONSUMPTION AND GROWTH: A FURTHER ANALYSIS, Akira Takayama.
164	THE PROBABILITY OF A CYCLICAL MAJORITY, Frank De Meyer and Charles R. Plott.

Paper No.	Title and Author(s)
166	THE CLASSROOM ECONOMY: RULES, RESULTS, REFLECTIONS, John A. Carlson.
167	AN ACTIVITY MODEL OF THE FIRM UNDER RISK, Carl R. Adams.
169	TAXES AND SHARE VALUATION IN COMPETITIVE MARKETS, Vernon L. Smith.
171	PROGRAMMING, PARETO OPTIMUM AND THE EXISTENCE OF COMPETITIVE EQUILIBRIA, Akira Takayama and Mohamed El-Hodiri.
178	ON THE STRUCTURE OF OPTIMAL GROWTH PROBLEM, Akira Takayama.
180	A NEW APPROACH TO DISCRETE MATHEMATICAL PROGRAMMING, G. W. Graves and Andrew B. Whinston.
181	EXPERIMENTING WITH THE ARMS RACE, Marc Pilisuk and Paul Skolnick.
186	REGIONAL ALLOCATION OF INVESTMENT: CORREGENDUM, Akira Takayama.
187	A SUGGESTED NEW MONETARY SYSTEM: THE GOLD VALUE STANDARD, Robert V. Horton.
193	MULTI-COMMODITY NETWORK FLOWS WITH MULTIPLE SOURCES AND SINKS, B. Rothchild and Andrew Whinston.
198	OPTIMAL DISPOSAL POLICIES, Carl Adams.
202	SOME FORMULAS ENCOUNTERED IN THE DEDUCTIVE ANALYSIS OF THIRD-ORDER AUTOGRESSION PROCESS, R. L. Basmann and R. J. Rohr.
215	A CONVERGENT PARETO-SATISFACTORY NON-TATONNEMENT ADJUSTMENT PROCESS FOR A CLASS OF UNSELFISH EXCHANGE ENVIRONMENTS, John O. Ledyard.
217	ON A "CONCAVE" CONTRACT CURVE, Akira Takayama.
218	THE EFFECTS OF FISCAL AND MONETARY POLICIES UNDER FLEXIBLE AND FIXED EXCHANGE RATES, Akira Takayama.
219	A MATCHING THEOREM FOR GRAPHS, D. Kleitman, A. Martin-Lof, B. Rothchild and A. Whinston.
224	GENERALIZED OPINION LEADERSHIP IN CONSUMER PRODUCTS: SOME PRELIMINARY FINDINGS, Charles W. King and John O. Summers.
226	THE FIRM AS AN AUTOMATION - I., Edward Ames.
227	SECOND-BEST SOLUTIONS, PEAK-LOADS AND MARGINAL COST PRICE POLICIES FOR PUBLIC UTILITIES, Robert A. Meyer, Jr.
228	EQUIPMENT REPLACEMENT UNDER UNCERTAINTY, Robert A. Meyer, Jr.

Paper No.	Title and Author(s)
233	ECONOMIC EFFECTS OF UNIFORM CONSUMER CREDIT CODE: A COMMENT, David C. Ewert.
234	OPTIMAL ADVERTISING EXPENDITURE IMPLICATIONS OF A SIMULTANEOUS-EQUATION REGRESSION ANALYSIS, Leonard J. Parsons and Frank M. Bass.
237	OPPOSITION OF PREFERENCES AND THE THEORY OF PUBLIC GOODS, Robert A. Meyer, Jr.
238	THE TAXATION OF RESTRICTED STOCK COMPENSATION PLANS, G. W. Hettenhouse and Wilbur G. Lewellen.
239	DECOMPOSABLE REGRESSION MODELS IN THE ANALYSIS OF MARKET POTENTIALS, Frank M. Bass.
241	OPPORTUNITY COSTS AND MODELS OF SCHOOLING IN THE NINETEENTH CENTURY, Lewis Solmon.
242	ESTIMATING FREQUENCY FUNCTIONS FROM LIMITED DATA, Keith C. Brown.
246	ON OPTIMAL CAPITAL ACCUMULATION IN THE PASINETTI MODEL OF GROWTH, S. C. Hu.
250	MONEY, INTEREST AND POLICY, P. H. Hendershott and George Horwich.
251	ON THE PEAK-LOAD PROBLEM, Akira Takayama.
253	A NOTE ON TECHNICAL PROGRESS, INVESTMENT, AND OPTIMAL GROWTH, Sheng Cheng Hu.
254	MANUFACTURERS' SALES AND INVENTORY ANTICIPATIONS: THE OBE COMPUTATIONAL PROCEDURES, John A. Carlson.
256	TWO ALGORITHMS FOR INTEGER OPTIMIZATION, Edna Loehman, Tuan Ph. Nghiem and Andrew Whinston.
260	AGE-DEPENDENT UTILITY IN THE LIFETIME ALLOCATION PROBLEM, Kenneth Avio.
261	AFFECTIVE AND VALUATIONAL CONSEQUENCES OF SELF-PERCEIVED UNIQUENESS DEPRIVATION: I. HYPOTHESES AND METHODOLOGICAL PRESCRIPTIONS, Howard Fromkin.
262	AFFECTIVE AND VALUATIONAL CONSEQUENCES OF SELF-PERCEIVED UNIQUENESS DEPRIVATION: II. EXPERIMENTALLY AROUSED FEELINGS OF SELF PERCEIVED SIMILARITY AS AN UNDESIRABLE AFFECTIVE STATE, Howard Fromkin.

Paper No.	Title and Author(s)
263	AFFECTIVE AND VALUATIONAL CONSEQUENCES OF SELF-PERCEIVED UNIQUENESS DEPRIVATION: III. THE EFFECTS OF EXPERIMENTALLY AROUSED FEELINGS OF SELF PERCEIVED SIMILARITY UPON VALUATION OF UNAVAILABLE AND NOVEL EXPERIENCES, Howard Fromkin.
264	AIR POLLUTION AND HOUSING: SOME FINDINGS, Robert J. Anderson, Jr., and Thomas D. Crocker.
265	APPLICATION OF REGRESSION MODELS IN MARKETING: TESTING VERSUS FORECASTING, Frank M. Bass.
267	A LINEAR PROGRAMMING APPROACH TO AIRPORT CONGESTION, Donald W. Kiefer.
268	ON PARETO OPTIMA AND COMPETITIVE EQUILIBRIA, PART I. RELATIONSHIP AMONG EQUILIBRIA AND OPTIMA, James C. Moore.
269	ON PARETO OPTIMA AND COMPETITIVE EQUILIBRIA, PART II. THE EXISTENCE OF EQUILIBRIA AND OPTIMA, James C. Moore.
271	A COMPARISON OF THREE MULTI-PRODUCT, MULTI-FACILITY BATCH SCHEDULING HEURISTICS, David R. Denzler.
272	A REPRESENTATION OF INTEGER POINTS IN POLYHEDRAL CONE, Ph. Tuan Nghiem.
273	LINE OF BUSINESS REPORTING - A METHODOLOGY FOR ESTIMATING BENEFITS, Russell M. Barefield.
274	MARKETING APPLICATIONS OF SELF-DESIGNATED OCCUPATION SKILL VARIABLES, E. A. Pessemier and D. J. Tigert.
275	THE FULL-EMPLOYMENT INTEREST RATE AND THE NEUTRALIZED MONEY STOCK, Patric H. Hendershott.
276	SOME APPLICATIONS OF THE CHANGE OF BASE TECHNIQUE IN INTEGER PROGRAMMING, Ph. Tuan Nghiem.
277	A WELFARE FUNCTION USING "RELATIVE INTENSITY" OF PREFERENCE, Frank DeMeyer and Charles R. Plott.
279	RACE AND COMPETENCE AS DETERMINANTS OF ACCEPTANCE OF NEWCOMERS IN SUCCESS AND FAILURE WORK GROUPS, Howard L. Fromkin, Richard J. Klimoski, and Michael F. Flanagan.
280	LEADERSHIP, POWER AND INFLUENCE, Donald C. King and Bernard B. Bass.
281	RECENT RESULTS IN THE THEORY OF VOTING, Charles R. Plott.
282	DISAGGREGATION OF ANALYSIS OF VARIANCE FOR PAIRED COMPARISONS: AN APPLICATION TO A MARKETING EXPERIMENT, E. A. Pessemier and R. D. Teach.

BEST COPY AVAILABLE

Paper No.	Title and Author(s)
283	MARKET RESPONSE TO INNOVATION, FURTHER APPLICATIONS OF THE BASS NEW PRODUCT GROWTH MODEL, John V. Nevers.
284	PROFESSIONALISM, UNIONISM, AND COLLECTIVE NEGOTIATION: TEACHER NEGOTIATIONS EXPERIENCE IN CALIFORNIA, James A. Craft.
285	A FREQUENCY DOMAIN TEST OF THE DISTURBANCE TERM IN LINEAR REGRESSION MODELS, Thomas F. Cargill and Robert A. Meyer.
286	EVALUATING ALTERNATIVE PROPOSALS AND SOURCES OF NEW INFORMATION, Edgar A. Pessemier.
287	A MULTIVARIATE REGRESSION ANALYSIS OF THE RESPONSES OF COMPETING BRANDS TO ADVERTISING, Frank M. Bass and Neil E. Beckwith.
288	ASSESSING REGULATORY ALTERNATIVES FOR THE NATURAL GAS PRODUCING INDUSTRY, Keith C. Brown.
289	TESTING AN ADAPTIVE INVENTORY CONTROL MODEL, D. Clay Whybark.
291	THE LABOR ASSIGNMENT DECISION: AN APPLICATION OF WORK FLOW STRUCTURE INFORMATION, William K. Holstein and William L. Berry.
294	AN EFFICIENT BRANCH AND BOUND ALGORITHM FOR THE WAREHOUSE LOCATION PROBLEM, Basheer M. Khumawala.
295	THE INTERACTION OF GROUP SIZE AND TASK STRUCTURE IN AN INDUSTRIAL ORGANIZATION, Robert C. Cummins and Donald C. King.
296	PROJECT AND PROGRAM DECISIONS IN RESEARCH AND DEVELOPMENT, Edgar A. Pessemier and Norman R. Baker.
298	SEGMENTING CONSUMER MARKETS WITH ACTIVITY AND ATTITUDE MEASURES, Thomas Hustad and Edgar Pessemier.
299	R & D MANAGERS' CHOICES OF DEVELOPMENT POLICIES IN SIMULATED R & D ENVIRONMENTS, Herbert Moskowitz.
300	DILUTION AND COUNTER-DILUTION IN REPORTING FOR DEFERRED EQUITY, Charles A. Tritschler.
301	A METHODOLOGY FOR THE DESIGN AND OPTIMIZATION OF INFORMATION PROCESSING SYSTEMS, J. F. Nunamaker, Jr.
303	ON PRODUCTION FUNCTIONS AND ELASTICITY OF SUBSTITUTION, K. R. Kadiyala.
304	AN EXPERIMENTAL INVESTIGATION OF DECISION MAKING IN A SIMULATED RESEARCH AND DEVELOPMENT ENVIRONMENT, Herbert Moskowitz.

Paper No.	Title and Author(s)
305	A NOTE ON MONEY AND GROWTH, Akira Takayama.
307	AN EXPERIMENTAL STUDY OF RELATIONSHIPS BETWEEN ATTITUDES, BRAND PREFERENCE AND CHOICE, Frank M. Bass, Edgar A. Pessemier, and Donald R. Lehmann.
309	WAGES AND HOURS AS SIGNIFICANT ISSUES IN COLLECTIVE BARGAINING, Paul V. Johnson.
311	AN EFFICIENT HEURISTIC ALGORITHM FOR THE WAREHOUSE LOCATION PROBLEM, Basheer M. Khumawala.
312	REACTIONS TO LEADERSHIP STYLE AS A FUNCTION OF PERSONALITY VARIABLES, M. H. Rucker and D. C. King.
313	FIRE FIGHTER STRATEGY IN WAGE NEGOTIATIONS, James A. Craft.
314	TESTING DISTRIBUTED LAG MODELS OF ADVERTISING EFFECT - AN ANALYSIS OF DIETARY WEIGHT CONTROL PRODUCT DATA, Frank M. Bass and Darrall G. Clarke.
316	AN EMPIRICAL INVESTIGATION OF THE RELIABILITY AND STABILITY OF SELECTED ACTIVITY AND ATTITUDE MEASURES, Edgar Pessemier and Albert Bruno
317	BEHAVIOR OF THE FIRM UNDER REGULATORY CONSTRAINT: CLARIFICATIONS, Mohamed El-Hodiri and Akira Takayama.
320	DEPRECIATION POLICY AND THE BEHAVIOR OF CORPORATE PROFITS, Russell M. Barefield and Eugene E. Comiskey.
321	LABORATORY RESEARCH AND THE ORGANIZATION: GENERALIZING FROM LAB TO LIFE, Howard L. Fromkin and Thomas M. Ostrom.
322	LOT SIZING PROCEDURES FOR REQUIREMENTS PLANNING SYSTEMS: A FRAMEWORK FOR ANALYSIS, William L. Berry.
326	PRIORITY SCHEDULING AND INVENTORY CONTROL IN JOB LOT MANUFACTURING SYSTEMS, William L. Berry.
328	THE EXPECTED RATE OF INFLATION BEFORE AND AFTER 1966: A CRITIQUE OF THE ANDERSEN-CARLSON EQUATION, Patric H. Hendershott.
330	A FURTHER PROBLEM IN LEAD-LAG DETECTION, Robert A. Meyer, Jr.
332	THE SMOOTHING HYPOTHESIS: AN ALTERNATIVE TEST, Russell M. Barefield and Eugene E. Comiskey.
333	CONSERVATISM IN GROUP INFORMATION PROCESSING BEHAVIOR UNDER VARYING MANAGEMENT INFORMATION SYSTEMS, Herbert Moskowitz.

Paper No.	Title and Author(s)
334	PRIMACY EFFECTS IN INFORMATION PROCESSING BEHAVIOR - THE INDIVIDUAL VERSUS THE GROUP, Herbert Moskowitz.
336	VEHICLE ROUTING FROM CENTRAL FACILITIES. Brian F. O'Neil and D. Clay Whybark.
339	UNEXPLAINED VARIANCE IN STUDIES OF CONSUMER BEHAVIOR, Frank M. Bass.
340	THE PRODUCTION FUNCTION AS A MODEL OF THE REQUIREMENTS OF THE INFANTRY SERGEANT'S ROLE, Richard C. Roistacher and John J. Sherwood.
341	SELECTING EXPONENTIAL SMOOTHING MODEL PARAMETERS: AN APPLICATION OF PATTERN SEARCH, William L. Berry and Friedhelm W. Bliemel.
342	AN INTEGRATED EXAMINATION OF MEDIA APPROACHES TO MARKET SEGMENTATION, Albert Bruno, Thomas Husted and Edgar Pessemier.
343	LABORATORY EXPERIMENTATION, Howard L. Fromkin and Siegfried Streufert.
344	REVERSAL OF THE ATTITUDE SIMILARITY-ATTRACTION EFFECT BY UNIQUENESS DEPRIVATION, Howard L. Fromkin, Robert L. Dipboye and Marilyn Pyle.
345	WILL THE REAL CONSUMER-ACTIVIST PLEASE STAND UP, Thomas P. Hustad and Edgar A. Pessemier.
346	MULTI-ATTRIBUTE MODELS FOR PREDICTING INDIVIDUAL PREFERENCE AND CHOICE, Edgar A. Pessemier.
347	THE VALUE OF INFORMATION IN AGGREGATE PRODUCTION PLANNING - A BEHAVIORAL EXPERIMENT, Herbert Moskowitz.
348	A MEASUREMENT AND COMPOSITION MODEL FOR INDIVIDUAL CHOICE AMONG SOCIAL ALTERNATIVES, Edgar A. Pessemier.
349	THE NEOCLASSICAL THEORY OF INVESTMENT AND ADJUSTMENT COSTS, Akira Takayama.
350	A SURVEY OF FACILITY LOCATION METHODS, D. Clay Whybark and Basheer M. Khumawala.
351	THE LOCUS AND BASIS OF INFLUENCE ON ORGANIZATION DECISIONS, Martin Patchen.
352	A PLEA FOR A FOURTH TRADITION - AND FOR ECONOMICS, Robert V. Horton.
353	EARLY APPLICATIONS OF SPECTRAL METHODS TO ECONOMIC TIME SERIES, Thomas F. Cargill.

Paper No.	Title and Author(s)
354	STUDENT APPLICATIONS IN A PRINCIPLES COURSE OF ECONOMIC ANALYSIS TO SELF-DISCOVERED ITEMS, Robert V. Horton.
355	BRANCH AND BOUND ALGORITHMS FOR LOCATING EMERGENCY SERVICE FACILITIES, Basheer M. Khumawala.
356	BEHAVIORAL SCIENCES LABORATORIES DESIGN FACTORS, Benjamin L. Mays.
357	AN EFFICIENT ALGORITHM FOR CENTRAL FACILITIES LOCATION, Basheer M. Khumawala.
358	AN EXPERIMENTAL STUDY OF ATTITUDE CHANGE, ADVERTISING, and USAGE IN NEW PRODUCT INTRODUCTION, James L. Ginter and Frank M. Bass.
359	DENIAL OF SELF-HELP REPOSSESSION: AN ECONOMIC ANALYSIS, Robert W. Johnson.
360	WAREHOUSE LOCATION WITH CONCAVE COSTS, Basheer M. Khumawala and David L. Kelly.
361	LINEAR AND NONLINEAR ESTIMATION OF PRODUCTION FUNCTIONS, R. A. Meyer and K. R. Kadiyala.
362	QUASI-CONCAVE MINIMIZATION SUBJECT TO LINEAR CONSTRAINTS, Antal Majthay and Andrew Whinston.
363	PRODUCTION FUNCTION THEORY AND THE OPTIMAL DESIGN OF WASTE TREATMENT FACILITIES, James R. Marsden, David E. Pingry and Andrew Whinston.
364	A REGIONAL PLANNING MODEL FOR WATER QUALITY CONTROL, David E. Pingry and Andrew Whinston.
365	ISSUES IN MARKETING'S USE OF MULTI-ATTRIBUTE ATTITUDE MODELS, William L. Wilkie and Edgar A. Pessemier.
366	A SOCIAL PSYCHOLOGICAL ANALYSIS OF ORGANIZATIONAL INTEGRATION, Howard L. Fromkin.
367	ECONOMICS OF WASTEWATER TREATMENT: THE ROLE OF REGRESSION, J. R. Marsden, D. E. Pingry and A. Whinston.
368	THE ROLE OF MODELS IN NEW PRODUCT PLANNING, Edgar A. Pessemier and H. Paul Root.
369	A NOTE ON PREFERENCE ORDERINGS WHICH ARE CONVEX TO THE ORIGIN, James C. Moore.
370	AXIOMATIC CHARACTERIZATIONS OF CONSUMER PREFERENCES AND THE STRUCTURE OF THE CONSUMPTION SET, James C. Moore.
371	BUSINESS POLICY OR STRATEGIC MANAGEMENT: A BROADER VIEW FOR AN EMERGING DISCIPLINE, Dan E. Schendel and Kenneth J. Hatten.

Paper No.	Title and Author(s)
372	MULTI-ATTRIBUTE CHOICE THEORY - A REVIEW AND ANALYSIS, Edgar A. Pessemier and William L. Wilkie.
373	INFORMATION AND DECISION SYSTEMS FOR PRODUCTION PLANNING: A NEED FOR AN INTER-DISCIPLINARY PERSPECTIVE, Herbert Moskowitz and Jeffrey G. Miller.
374	ACCOUNTING FOR THE MAN/INFORMATION INTERFACE IN MANAGEMENT INFORMATION SYSTEMS, Herbert Moskowitz and Richard O. Mason.
375	A COMPETITIVE PARITY APPROACH TO COMPETITION IN A DYNAMIC MARKET MODEL, Randall L. Schultz.
376	BEHAVIORAL MODEL BUILDING, Randall L. Schultz and Dennis P. Slevin.
377	THE HALO EFFECT AND RELATED ISSUES IN MULTI-ATTRIBUTE ATTITUDE MODELS - AN EXPERIMENT, William L. Wilkie and John M. McCann.
378	AN IMPROVED METHOD FOR SOLVING THE SEGREGATED STORAGE PROBLEM, Basheer M. Khumawala and David G. Dannenbring.
379	ON THE PROBABILITY OF WINNING IN COMPETITIVE BIDDING THEORY, Keith C. Brown.
380	COST ALLOCATION FOR RIVER BASIN PLANNING MODELS, E. Loehman, D. Pingry and A. Whinston.
381	FORECASTING DEMAND FOR MEDICAL SUPPLY ITEMS USING EXPONENTIAL AND ADAPTIVE SMOOTHING MODELS, Everett E. Adam, Jr., Willian L. Berry and D. Clay Whybark.
382	SETTING ADVERTISING APPROPRIATIONS: DECISION MODELS AND ECONOMETRIC RESEARCH, Leonard J. Parsons and Randall L. Schultz.
383	ON THE OPTIMAL GROWTH OF THE TWO SECTOR ECONOMY, John Z. Drabicki and Akira Takayama.
384	UNCERTAIN COSTS IN COMPETITIVE BIDDING, Keith C. Brown.
385	EFFECTS OF THE NUMBER AND TYPE OF ATTRIBUTES INCLUDED IN AN ATTITUDE MODEL: MORE IS NOT BETTER, William L. Wilkie and Rolf P. Weinreich.
386	PARETO OPTIMAL ALLOCATIONS AS COMPETITIVE EQUILIBRIA, James C. Moore.
387	A PLANNING AND COST ALLOCATION PROCEDURE FOR COMPUTER SYSTEM MANAGEMENT, J. F. Nunamaker and A. Whinston.
388	PROFESSOR DEBREU'S "MARKET EQUILIBRIUM" THEOREM: AN EXPOSITORY NOTE, James C. Moore.

Paper No.	Title and Author(s)
389	THE ASSIGNMENT OF MEN TO MACHINES: AN APPLICATION OF BRANCH AND BOUND. Jeffrey G. Miller and William L. Berry.
390	THE IMPACT OF HIERARCHY AND GROUP STRUCTURE ON INFORMATION PROCESSING IN DECISION MAKING: APPLICATION OF A NETWORKS/ SYSTEMS APPROACH, David L. Ford, Jr.
391	PROCESSING SYSTEMS OPTIMIZATION THROUGH AUTOMATIC DESIGN AND REORGANIZATION OF PROGRAM MODULES, J. F. Nunamaker, Jr., W. C. Nylin, Jr. and Benn Konsynski.