IPSS/DBMS: A SIMULATOR FOR MODELING DATA BASE MANAGEMENT SYSTEMS*

Thomas G. DeLutis and Joseph D. Smith
The Department of Computer and Information Science
The Ohio State University

ABSTRACT

A special purpose discrete event simulator has been developed to investigate the behavior of network and hierarchical classes of data base management systems. It has been named IPSS/DBMS (Information Processing System Simulator/Data Base Management System). The purpose of this paper is to 1) present the salient features of IPSS/DBMS, 2) relate them to the current DBMS data structure and software architectures, and 3) outline the basic approach to model synthesis using IPSS/DBMS. Features described are the mechanism employed to define and measure schema, schema to schema translations, DBMS software and query processing activities. The interaction between IPSS/DBMS and the standard IPSS simulator is also discussed.

INTRODUCTION

A data base management system (DBMS) is a collection of software procedures which facilitates access to a data base[1] which is shared by diverse users. The primary aim of the analyst/designer is to define a schema which is satisfactory to all users. A secondary goal is the definition of restricted views of the data base which are particular to the needs and access rights of the user, i.e., to define subschemas. A simulator has been formulated to aid the analyst/designer in these endeavors. It has been specifically designed for investigating the behavior of DBMSs with respect to the user demands, the data base contents, the schema and subschema formulations, and the DBMS software. Attention is focused on the appropriateness of a DBMS's schema and subschema structures with respect to the data base contents and to various system loadings.

This simulator provides the analyst/designer with facilities for characterizing the salient features of modern data base management systems. Language constructs that are included enable the modeler to define the following for a given DBMS system:

1. data structure,
2. data base content,
3. data manipulation language operations,
4. task management and resource allocation policies,
5. interfaces to the application software and the operating system,
6. translation algorithms between logical views of the data base,
7. performance measures for identifying the behavior of the DBMS.

This simulator is not based on a specific DBMS data structure. However, the language constructs favor those DBMSs commonly classified as having hierarchical or network-like data structures. Representative of such systems are respectively IBM's Information Management System (10) and CODASYL's Data Description Language Committee recommendations (3).

AN OVERVIEW OF DBMS FUNCTIONS

The evolution from sequential, batch environments to on-line information systems has been facilitated by rapid advances in hardware and software technologies. However, the impetus has come from increasing demand for timely data in support of management and administrative processes, and thus the need for integrated views of an organization's data base. Data base management systems are a natural outgrowth of the increased demands for system development and implementation efficiencies. While on-line integrated data bases provide the analyst with a means for decreasing design and programming efforts, they also have the potential for substantially decreasing the overall executional efficiency of the information system. Critical factors affecting executional efficiency are both definitional and operational in nature; not all of which may be under the designer's control. However, they must be accounted for in any methodology attempting to model DBMS behavior. These factors include:

[1]A data base is defined to be the collection of data records known to the DBMS plus the a priori defined inter-record linkages.

* This research is sponsored in part by the National Science Foundation, Grant No. SIS75-21648.

I. Definitional Factors

A. Data Structure--the formal mechanism for partitioning the data base as a function of data attributes and the assumed needs of user query (or application) processing.

B. Data Manipulation Operations--the operations available for traversing an occurrence of a data structure as a function of known data attributes.

C. Subschema - Schema Mappings--the identification of a file's data origins in terms of corresponding files at the next, more primitive level of data base definition.

D. DBMS Resources--the hardware and software resources controlled by the DBMS. Included are main memory used as record buffers (or work areas), and software procedures invoked to perform needed I/O services.

E. DBMS Management Policies--system software responsible for resource scheduling and task management.

II. Operational Factors

A. Data Base Contents--the actual data contained in the information systems data base. Important parameters include the volume of data and its distribution of attributes.

B. System Loading--the volume of activity against the data base generated by the system users. Volume must be weighted by query type and arrival patterns.

Ghosh and Tuel (7) experimented with three of these factors on an IMS data base. They found that the sequence of data base accesses did not have a significant effect on access time but that the data structure (logical and physical) and the data base contents did. Their experiments involved the use of the data base and corresponding hardware on a stand-alone basis. While this situation is desirable, it is frequently impossible to obtain. While several performance evaluation methodologies and DBMS simulations have been previously suggested (9, 12, 13, 14, 15), none can address all the DBMS critical factors listed above.

The simulator described in this paper has been developed based on the following functional behavior for a DBMS. All activities occur in response to simuli (referred to as a request for service); for example, application software executes in response to system user requests and, as part of their execution, create DBMS requests by invoking DML operations. The DBMS services this stream of DML requests generated by the application. These requests represent access to logical data bases known to each application and can require either the retrieval of existing data, the insertion of new data, or both. The important point is that the requests are at the logical application level which provides application independence and portability. The data base may have many levels each with its corresponding DML operations. Multilevel views of DBMS data help achieve data independence by insulating the application program from the physical data access mechanisms and the storage structure. The CODASYL Data Base Task Group Report (2) and the Joint GUIDE-SHARE Data Base Requirements Group Report (8) generated the basis for much of the subsequent discussion and understanding of this notion. In addition, other reports have also described logical data views where each succeeding level represents a more detailed view of the actual data base storage and access capabilities (1, 4, 6, 11).

As a result of multi-level DBMS description, the DBMS activities which are performed in response to a DML invocation by the application may itself be expanded into additional, more primitive DML operations which reference other views of the data base. The following is the assumed general behavior of the DBMS for a single invocation of a DML (4). The application repeats this process as it services a user query. This discussion is based on a two-level data base structure having a global schema and one or more local subschemas.

1. The application initiates a DML operation (A) in order to reference logical record (1), i.e., $DML_A(L)$ in Figure 1.

2. The DBMS uses the subschema definition to expand the DML operation and referenced logical record into a series of more primitive DML operations and an associated sequence of logical records:

$$DML_A(L) \Rightarrow \{DML_{a_1}(L_1),\ldots,DML_{a_i}(L_j),\ldots, DML_{a_k}(L)\}.$$

This sequence defines the activity required of the DBMS to obtain the desired record based upon the subschema data structure and its knowledge of previous references to this level data base.

3. The DBMS uses the subschema-schema translation algorithms which are specific to the identified subschema to transform this application oriented DML sequence into a sequence of DBMS oriented DML operations (D) referencing its own logical records (R):

$$DML_{a_i}(L_j) \Rightarrow \{DML_{D_{i1}}(R_{j1}),\ldots,DML_{K_{i\ell}}(R_{jm}),\ldots, DML_{D_{in}}(R_{jo})\}.$$

The generation of this sequence occurs in two phases; phase one translates the logical record L_j into those schema records containing its data. This is the definition of the subschema record in terms of one or more schema records. Notationally, the translation is:

$$L_j \Rightarrow \{R_{j1}, R_{j2}, \ldots, R_{jm}, \ldots, R_{jo}\}.$$

Phase two is the translation of the subschema data manipulation language command,

$$DML_{a_i}(\{R_{j1}, \ldots, R_{jm}, \ldots, R_{jo}\})$$

into the appropriate schema DML sequence.

4. The sequence of operations and record references generated in step three is expanded to reflect the interaction of the DBMS DMLs with its subschema definition. This step is similar in operation to step 2 above, i.e., each DML generated in step 3 is expanded into a sequence of data base references as a function of the schema data structure and previous references:

$$DML_{D_{i\ell}}(R_{jm}) \Rightarrow \{DML_{d_1}(R_{jm1}), \ldots, DML_{d_p}(R_{jmq}), \ldots, DML_{d_n}(R_{jm})\}.$$

5. The DBMS sequence is translated to a sequence of references to the storage data base. Records referenced at this level are those belonging to the system's data sets whose organization methods are assumed to be those recognized by conventional input/output control systems and whose DML operations represent the system's file access method algorithms. Notationally, the transformation is:

$$DML_{d_p}(R_{jmq}) \Rightarrow \{DML_{IO_1}(P_{jmq1}), \ldots, DML_{IO_s}(P_{jmqt}), \ldots, DML_{IO_u}(P_{jmqu})\}.$$

Thus, translation is analogous to the one performed in step 3. Here, phase one operates to translate the schema record into its corresponding physical data base records:

$$R_{jmq} \Rightarrow \{P_{jmq1}, P_{jmq2}, \ldots, P_{jmqt}, \ldots, P_{jmqu}\},$$

and phase two translates the data manipulation language commands into the required sequence of file accesses.

As illustrated in Figure 1, in addition to the downward migration required to access the physical data base, there is a corresponding upward migration of data through various work areas. Thus, the DBMS must perform a high degree of processing and record keeping in order to provide application independence from the physical data base. However, performance criteria dictate that the information system analyst/designer be aware of the potentially high overhead incurred in these systems and that the DBMS design be as efficient as is practical within the operating constraints. Obviously, the DBMS provides many functions in addition to those outlined above to ensure data integrity, data base security and proper sequencing of operations. The paper does not address these problems, instead it concentrates on the DML and data migrations outlined above.

MODELING FACILITIES

Ascertaining the performance of a DBMS requires the characterization of definitional and operational factors, and the evaluation of their interaction under system loading. Thus, the simulator employs two basic constructs, one static in nature to define the desired schemata and their associated data structures and the second procedural in form to define the DMLs, the schema to schema transformations, and the query processing activities. The simulator has been structured so that model synthesis is modular in order to minimize the effort in evaluating alternate designs and to speed the evaluation process. Furthermore, the simulator can be used as a standalone system or in conjunction with the Information Processing System Simulator (IPSS)[2] to provide a more complete model of a total information system (5). Because DBMSs are functionally similar to operating systems, many of the features of IPSS which are used to describe operating systems have been incorporated into this simulator, thus it is an upward extension to this previous effort, and will be referred to as IPSS/DBMS.

Central to the construction and execution of a model are procedural facilities called services. These facilities model the system's software resources and, in their execution, reference the nonprocedural schema and data structure definitions and also invoke other services. The simulator provides a set of special language statements to assist in modeling DBMS functions via the service definitions. Figure 2 illustrates the essential character of a service facility. A service when executed performs the following functions:

1. Requests hardware and software resources needed for its execution, e.g., memory buffers, and other software modules.
2. Sequences the subordinate services which are invoked as part of its processing by initiating and waiting for their completion. These services are defined using the following IPSS/DBMS statements:
   a. Other Service Facilities:

      INITIATE SERVICE and
      WAIT SERVICE COMPLETE

   b. DML Service Facilities:

      INITIATE DML and
      WAIT DML COMPLETE

[2]IPSS is a special purpose discrete event digital simulation package designed specifically to investigate the behavior of more conventional information processing systems.

This form of service call causes the simulator to generate performance statistics for the named DML operation.

c. Schemata Translation Service Facilities:

ACQUIRE RECORD
WAIT RECORD AVAILABLE
STORE RECORD and
WAIT OPERATION COMPLETE

Here, the service calls are implicit, that is, the service facility defining the translation is referenced indirectly through the record type definition. Again, statistics are automatically gathered for the translation activities.

3. Processes records executing the WAIT PROCESS TIME statement which specifies the aggregate processing time.[3]

4. Releases acquired resources upon the completion of a service.

In addition, services have system supplied prologues and epilogues which result in either reentrant or serially reusable software procedures.

The second set of basic definitional constructs apply to the characterization of schemata and their associated record types; they are

DEFINE RECORD TYPE
DEFINE SET and
DEFINE SCHEMA

Within a simulation model these definitions characterize the data base as viewed by different processing units (e.g., the application, the DBMS, etc.). The role of each of these statements follows:

1. DEFINE RECORD TYPE This statement defines collections of logical records which have similar purpose and format. Included in a record type definition are parameters which characterize:

   a. the record format,

   b. the number of occurrences in the collection,

   c. the access characteristics (i.e., sequential or direct), and

   d. the linkage mechanism between record occurrences.

   In addition, this statement identifies associated service facilities to be invoked when a record occurrence is acquired or stored.

2. DEFINE SET This statement defines a named association among record types. The defined sets are restricted to a single schema. The set characterizes a relationship between two record types. This characterization includes:

   a. the name of the owner and member record types,

   b. the name of a procedure which distributes member record occurrences to owner occurrences,

   c. a definition for the linkage mechanisms used to traverse the set, and

   d. the number of set occurrences.

3. DEFINE SCHEMA The Define Schema statement allows the modeler to define a network of sets representing one of the DBMS's schemata. This definition is used by the IPSS/DBMS system as a template so that schema definitions can be tailored to the particular needs of one or more applications during a simulation. These definitions are referenced by the FIND operation which gathers statistics measuring the efficiency of the schema with respect to system activity. The outputs of the FIND can be used to drive the DML operations of the model.

[3]The standard simulation clock unit is tenths of milliseconds, however, the simulator allows this to be scaled.

MODEL SYNTHESIS

This section describes the use of the definitional facilities discussed in Section 3 to model DBMS behavior as described in Section 2. Figure 3 illustrates the basic approach for synthesizing models of a DBMS. While Figure 3 shows two schemata (i.e., the applications' subschema and the DBMS schema), the simulator is applicable to a broad spectrum of DBMS organizations.

Characterization of Application Programs

An application's behavior is viewed as the transformation of a query into one or more accesses (via DMLs) to the data base. The data base can be a physical one, or a logical one defined by a subschema. In an IPSS/DBMS simulation model, the application is a service facility definition, and the DMLs are other service facility definitions. The modeler can refer to a subschema definition during the execution of the application service facility for the purpose of determining the sequence of DMLs to be initiated. The subschema definition is treated as a template and therefore many realizations can be created and destroyed dynamically during the course of a simulation. The creation of a subschema realization represents that part of the total data base known by the application with respect to particular query, i.e., it represents the data base's data content and data structure. The FIND command assists the modeler in traversing a specific subschema realization, and traversal based performance statistics are automatically gathered by the execution of this command.

Characterization of Subschema DML Statements

The purpose of subschema DMLs is to effect the desired access to the data base as viewed through its subschema. The basic functions of an application are the retrieval and storage of data. The behavior of a DML with respect to these functions depends upon the operation desired, the data structure used to connect records, the location and state of a record (i.e., is it in a buffer, is it being multiply accessed, etc.), and the relation of the record to the last access to the data base. Since there are a broad spectrum of possible DMLs the simulator makes no attempt to predefine these routines. Instead it treats the DMLs as service facilities whose definitions are left to the modeler. However, the IPSS/DBMS system has extensive user library and macro features which enable the modeler to predefine these service facilities for later referal.

The DML completes its service when the desired records become available in the application's work area (i.e., has been acquired) or has been written to physical secondary storage (i.e., stored). However, the materialization of a record may require considerable data access activity at the schema and lower data base levels. Therefore, the request for a specific logical record is treated in the simulation as an implicit invocation of a service facility responsible for its materialization. The identity of the service facility is part of the record type definition and is invoked by the execution of the IPSS/DBMS ACQUIRE RECORD command.

Characterization of Subschema-Schema Translation

The translation procedures operate in a manner analogous to an application, except that instead of processing a query, it transforms a subschema record request into one or more data base accesses (via schema level DMLs). The data base can be a physical one, or a logical one defined by a schema. Other than the schema and DMLs referenced, the service facility operates in the same manner as described above for application programs.

Characterization of Schema DML Statements

The purpose of this level of DML is to access the schema data base. Service facility definitions have the same definitional characteristics as above with the exception of the facilities they reference.

Characterization of Schema IOCS Translation

These procedures are the interface between the DBMS and the underlying operating system. They translate DBMS schema records into actual system files accesses. To do this, these service facilities generate a sequence of calls to the underlying system. Two methods are available for modeling the system's behavior. One treats the system in the aggregate, i.e., as a black box while the second causes the IPSS/DBMS simulator to communicate with an independent simulation, written in IPSS, which is a model of the underlying system.

Computer System Characterization

A DBMS is a subsystem within a more general processing system and this is reflected in the IPSS/DBMS simulator. Characterization of the remaining components of an information processing system is accomplished in a second simulation model written in IPSS. DBMS calls to the operating system become invocations of service facilities in the IPSS model. Figure 4 illustrates the general interrelation between IPSS/DBMS and basic IPSS. Either system is capable of executing without the other, thereby providing the modeler with increased modeling flexibility.

PERFORMANCE MEASURES

The goal of the IPSS/DBMS simulator is to provide performance measures specific to DBMS behavior which will aid the analyst in his evaluation efforts. Performance can be measured using many criteria and from many points of view. The IPSS simulator provides measures associated with secondary storage system resources (i.e., queueing, resource utilization and activity levels) or operating system (queuing, task and resource management). Here the measures relative to resource behavior have been retained and new statistics specific to data base schemata definitions have been added. The purpose of these measures is to identify the behavior characteristics of the DBMS as a function of a schema definition and the data base usage.

Measures Related to Resources

Associated with each resource are a set of queueing and utilization statistics. Included are the following:

1. Queueing Statistics
   a. Busy period statistics--indicate the distribution of queueing activity within simulated time,
   b. Idle period statistics--indicate the distribution of queue idleness,
   c. Queue length statistics--measures aggregate queue behavior,
   d. Queue entry statistics--measures the volume of activity, and
   e. Queue transit time statistics--measures transit behavior through the queue.
2. Utilization Statistics
   a. Busy period statistics--indicate the distribution of queueing activity within simulated time,
   b. Idle period statistics--indicate the distribution of queue idleness,
   c. Concurrency statistics--provide a measure of concurrent service,
   d. Seizure statistics--measures the volume of activity for the resource, and
   e. Service time statistics--measures aggregate resource allocation to tasks or time to service requests.

Measures Related to Tasks

A task is viewed as the basic unit within an information processing system to which resources can be allocated and therefore is the mechanism used for accounting purposes. In IPSS/DBMS the modeler can associate activities defined by service facilities with one or more task facilities. The purpose of the task facility is to provide the modeler with a mechanism for distributing service facility performance statistics to user related modeling activities. Task statistics are gathered automatically throughout the simulation and are printed at the end of simulation. The task facility is general in construct and can be defined at any level in the modeled system. Therefore, this mechanism provides the modeler with a statistics gathering capability for measuring any level of DBMS activity. The statistical outputs are similar to the queueing and utilization statistics described above.

SUMMARY

IPSS/DBMS is a discrete event digital simulator specifically designed to aid the designer/analyst in investigating the behavior of data base management systems. The simulator provides special language constructs which enable the modeler to define the salient features of a wide spectrum of data base management systems of the hierarchical and network variety. The language constructs focus on the definition of schemata and on the description of DBMS behavior with respect to information system loading. The assumption is that DBMSs are adjunct software to the standard operating system. Therefore, this simulator focuses only on the DBMS level of total information system activity. However, the simulator has been designed in a modular manner so that it can easily interface with the standard IPSS. The analyst can thus use the IPSS/DBMS facilities to develop models of data base management systems independently of a detailed characterization of the underlying physical system, and use the standard interfaces between these two simulators for a more complete analysis of information system behavior.

BIBLIOGRAPHY

1. ANSI/X3/SPARC Study Group on Data Base Management Systems: Interim Report, (8 Feb 1975).

2. CODASYL Data Base Task Group, April 1971 Report, (available from ACM).

3. CODASYL Data Description Language Committee, CODASYL Data Description Language Journal of Development, (June 1973), NBS Handbook 113, (January 1974).

4. DeLutis, T.G., "Data Base Management Systems: Structures and Data Access", Proceedings of Conference on Energy Related Modeling and Data Base Management. Brookhaven National Laboratory, Upton, L.I., New York, 1975.

5. DeLutis, T.D., "The Information Processing Simulator - IPSS", Proceedings of the Ninth Annual Simulation Symposium, Tampa, Florida, 1976.

6. Fogt, K.J., "Data Structure Levels in Info. Systems", NTIS Report AD-764 954, 1973.

7. Ghosh, S.P., and Tuel, W.G., "A Design of An Experiment to Model Data Base System Performance", IEEE Transactions on Software Engineering, SE-2,2 (1976), 97-106.

8. GUIDE-Share, Data Base Requirements Group Report. November, 1970.

9. Hall, W.A., "A Simulation Model to Aid in the Design and Tuning of Hierarchical Data Bases", Winter Simulation Conference, 1974, 277-284.

10. Information Management System/360, Version 2, General Information Manual, GH20-0765, 1973, IBM Corporation, While Plains, New York.

11. McGee, W.C., "Some Current Issues in Data Description", Proceedings of ACM SIGFIDIT Workshop on Data Description, Access and Control, 1972, 1-12.

12. Reiter, A., "Data Models for Secondary Storage Reprentation", MRC Technical Summary Report #1554, Mathematics Research Center, University of Wisconsin - Madison, 1975.

13. Reiter, A., "Some Experiments in Directory Organization -- A Simulation Study", MRC Technical Report #1608, Mathematics Research Center, University of Wisconsin - Madison, 1975.

14. Sable, J.D., "Generalized Data Management Systems: Query and Language Processing", Proceedings of the International Seminar on Information Storage and Retrieval, 1971, 93-111.

15. Scheuermann, P., and Heller, J., "A View of Logical Data Organization and its Mapping to Physical Storage", Technical Report #38, Department of Computer Science, State University of New York at Stony Brook, 1974.

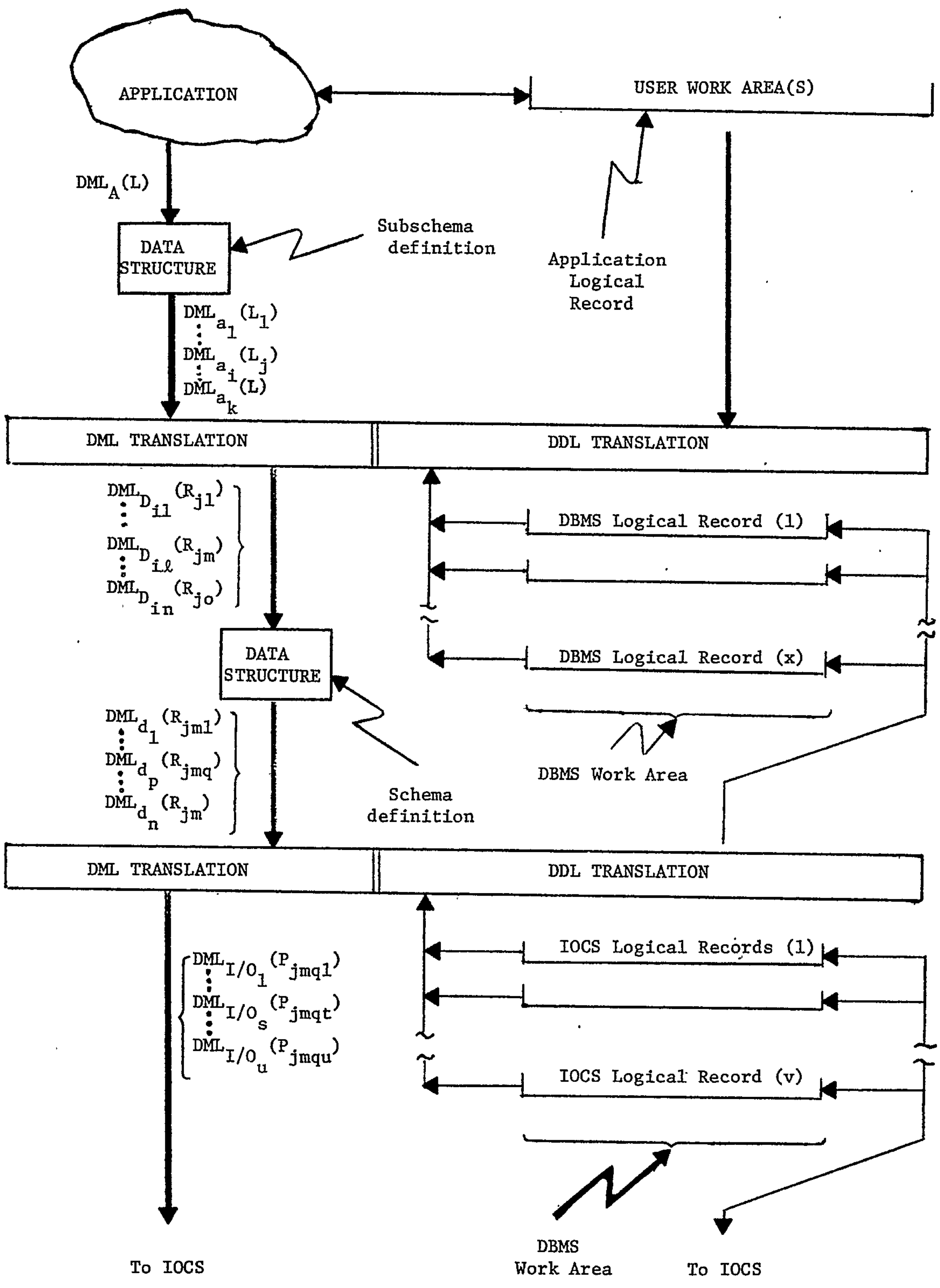
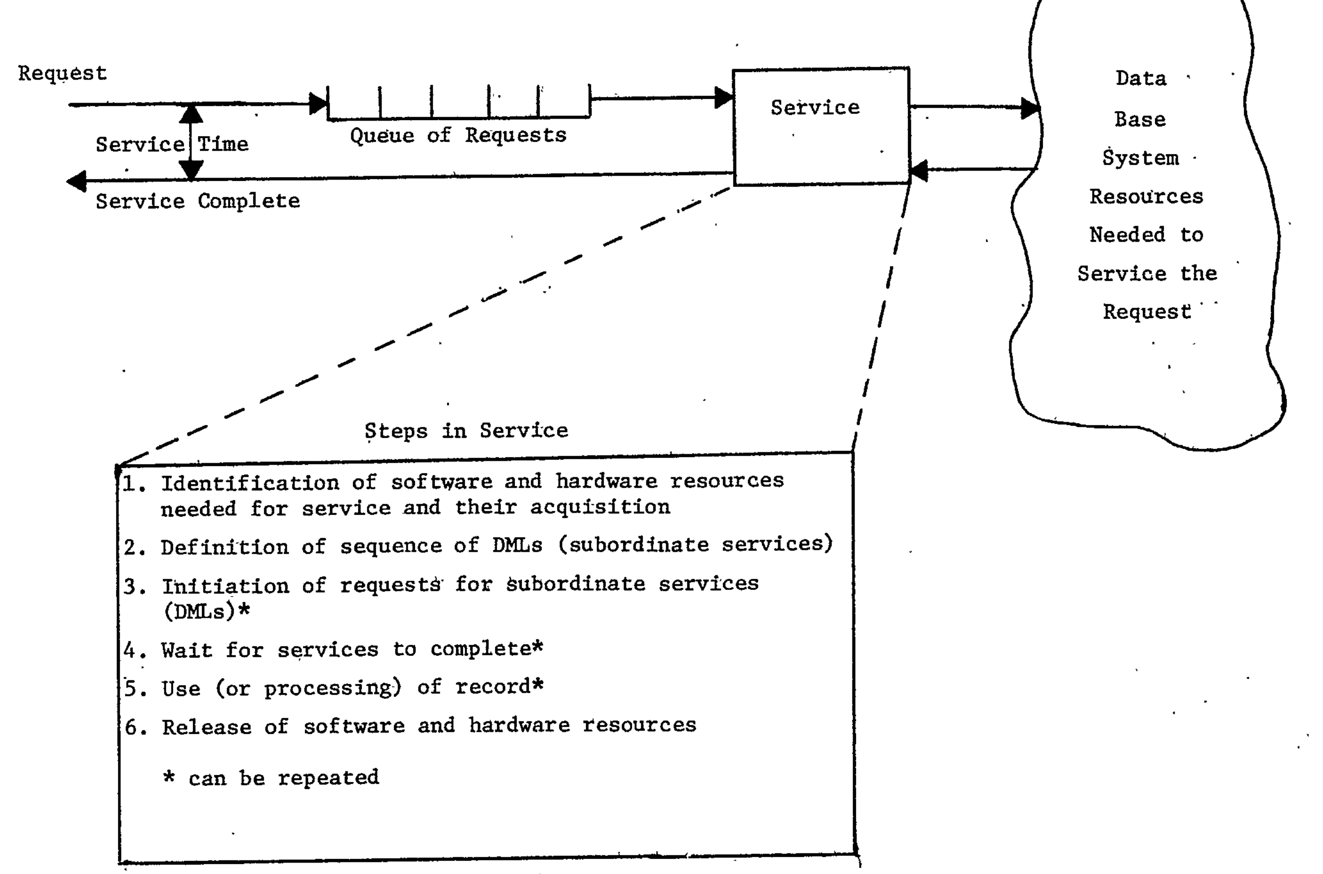


FIGURE 1

Functional Operation of the Data Base Management System

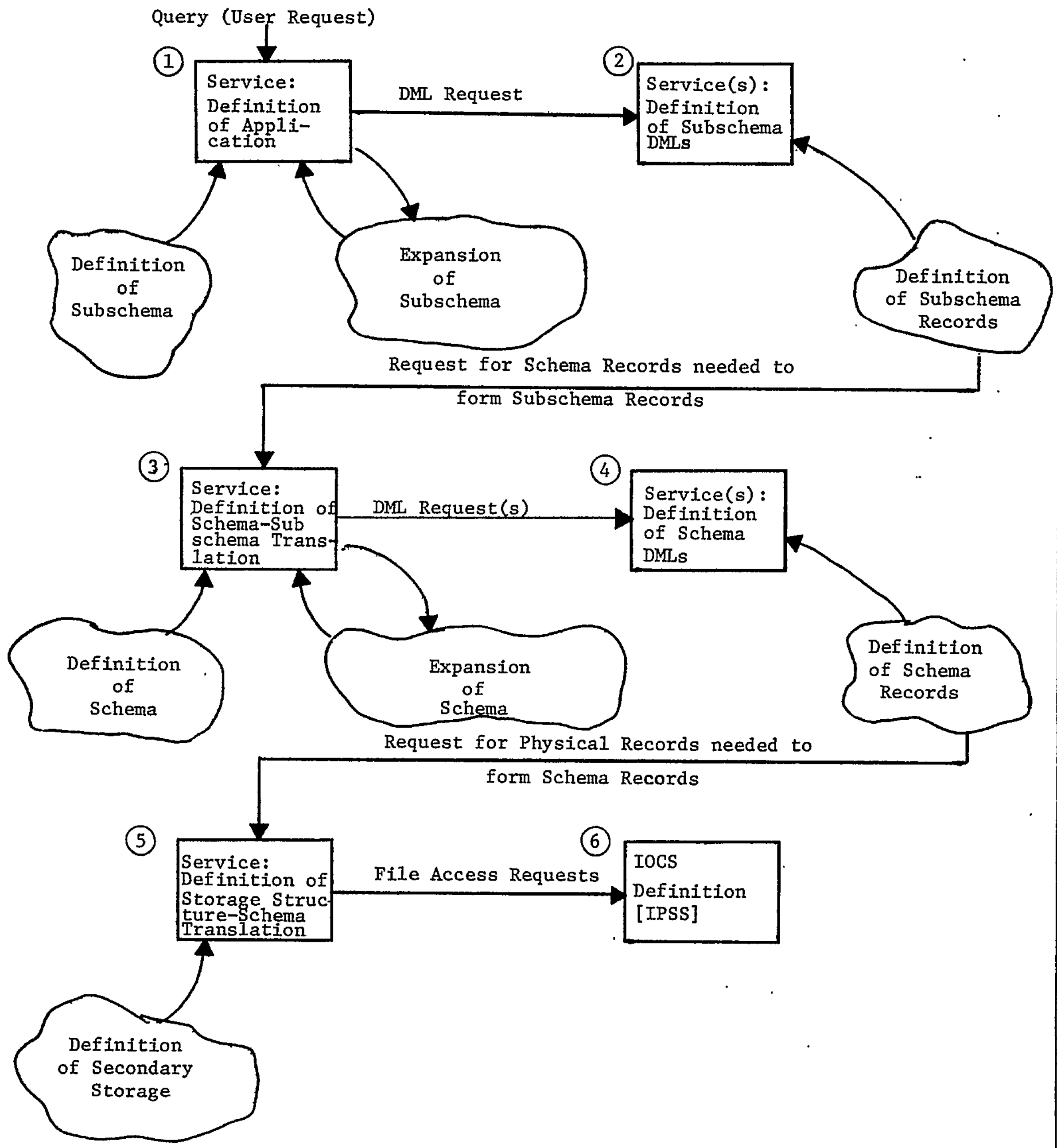
FIGURE 2
Service Facility Definition and Execution



1. Identification of software and hardware resources needed for service and their acquisition
2. Definition of sequence of DMLs (subordinate services)
3. Initiation of requests for subordinate services (DMLs)*
4. Wait for services to complete*
5. Use (or processing) of record*
6. Release of software and hardware resources

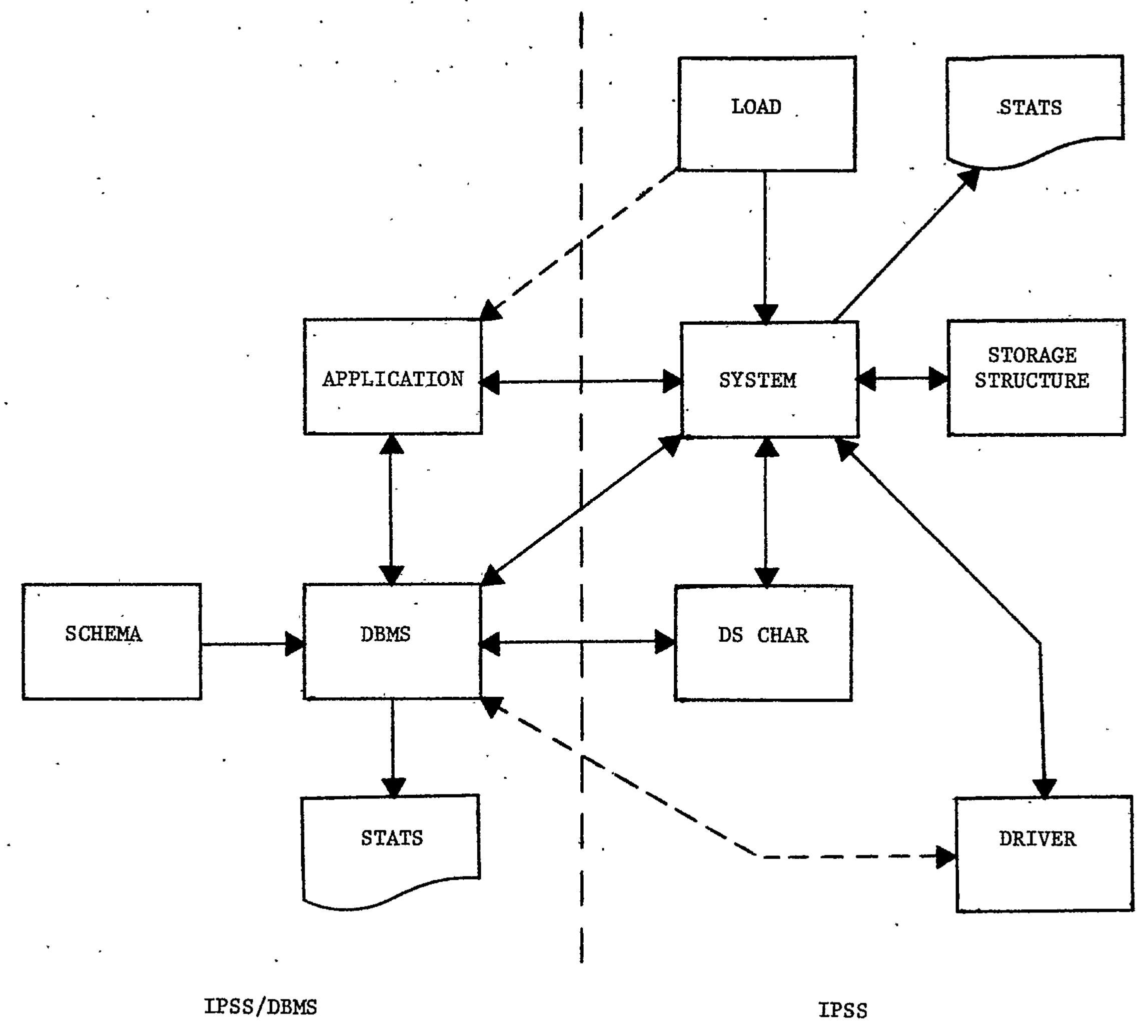
* can be repeated

FIGURE 3
Functional Behavior of IPSS/DBMS Model



Query (User Request)
1
Service: Definition of Application
DML Request
2
Service(s): Definition of Subschema DMLs
Definition of Subschema
Expansion of Subschema
Definition of Subschema Records
Request for Schema Records needed to form Subschema Records
3
Service: Definition of Schema-Sub schema Translation
DML Request(s)
4
Service(s): Definition of Schema DMLs
Definition of Schema
Expansion of Schema
Definition of Schema Records
Request for Physical Records needed to form Schema Records
5
Service: Definition of Storage Structure-Schema Translation
File Access Requests
6
IOCS Definition [IPSS]
Definition of Secondary Storage

FIGURE 4
The Interrelation Between IPSS/DBMS and IPSS


LOAD
STATS
APPLICATION
SYSTEM
STORAGE STRUCTURE
SCHEMA
DBMS
DS CHAR
STATS
DRIVER
IPSS/DBMS
IPSS