DATABASE SYSTEMS AS CONTROLLERS, MANAGERS AND LINGUISTS

A STUDY OF THE RELATIONSHIP OF DATABASE AND OPERATING SYSTEMS*

David K. Hsiao
Department of Computer Science
Naval Postgraduate School
Monterey, California 93943
U. S. A.

ABSTRACT

The study of the relationship of the database system and the operating system can best be characterized in terms of their issues on resource management and language support. In this paper the similarities and differences of these two types of systems (i.e., database systems and operating systems) on resource management and language support are articulated. It is hoped that this articulation can point out the new and future database-system research and development areas.

1. INTRODUCTION

Modern database systems can be viewed in terms of their capabilities in managing the databases and in supporting the data models and the data languages. This view has been a recent one. We first examine the change of a database system from an application program to a database manager and controller in Section 2. This change effects the relationship of the database and the operating systems. We then examine the evolution of database system from a mono-lingual and mono-model system to a multi-lingual and multi-model system in Section 3. This evolution has its precedence and similarity in the evolution of the operating system. We may gain considerable perspective by also reviewing the evolution of the operating system in this context. Lastly, we offer some concluding remarks in Section 4.

2. THE DATABASE SYSTEM AS A MANAGER AND CONTROLLER

The operating system has been the manager of the computer's physical resources, such as processors, processes, data sets, memories and devices. Physical resources do not have built-in relationships. For example, a blank tape has no bearing on a loaded disk. Consequently, the operating system schedules and controls the use of resources without requiring to interpret them and to maintain their relationships. The database system, on the other hand, manages logical resources such as relationships of data aggregates, attributes of data items and types of data values. Similarly, the database system schedules and controls the use of these logical resources (called databases). However, unlike the operating system, the database system must interpret the databases by way of their data models and maintain them at the direction of the data languages.

Although it is a resource manager, the database system has, in the past, been regarded by the operating system of the mainframe as an application program. In fact, an operating system of the mainframe also treats all of the other software systems as applications. Thus, we have, for example, the compiler system as an application. The compiler system produces compiled programs for execution. As long as these programs can be executed efficiently, the slow and batch-oriented compilation process may be tolerated. The database system is required to provide on-line inquiry and interactive sessions. The database transactions are not always carefully prepared and previously written programs. They usually consist of ad-hoc queries and interactive dialogues. They cannot be executed efficiently as application programs, since they require considerable direct control and support of the physical resources.

Modern database systems are either single-backend oriented or multiple-backend oriented. A single-backend database system utilizes a dedicated computer and specially designed database operating system so that database transactions can be executed in the backend efficiently and effectively. Furthermore, the database resides on the disks of the backend for more efficient access and control. In this approach, the mainframe is devoid of database-system software and databases. Like the communications frontend of a mainframe, which handles the communications efficiently and cost-effectively, the single-backend database system becomes the backend of the mainframe for cost-effective and efficient database management and control. The pioneering research work of XDMS* and the commercial work of Briton-Lee IDM-500 are excellent examples of the single-backend database systems.

For even more performance gains and capacity growth, there are the multiple-backend database systems. A multiple-backend database system is a parallel database system with a varying number of processors (called backends) and their disk subsystems. Modern multiple-backend database systems can relate the response-time improvement with the number of backends used. In fact, the performance gains in response-time improvement is nearly in inverse proportion to the number of backends added. For example, if we double the number of backends we may cut the response times for the same transaction of the same database by half. Obviously, the database (operating) system of the multiple-backend database system does not resemble either the mainframe-based or the single-backend database system. Specializations in database (operating) system development are mandatory requirements in such a highly-parallel database system. Examples of the multiple-backend database system can be found in the experimental multi-backend database system (MBDS)[14][15][16] and the commercial Teradata DBC/1013[17].

In conclusion, for high-performance and great-capacity database systems one should not rely on the general-purpose operating system and on the traditional mainframes. One should rely on a specially designed database (operating) system and on stand-alone, dedicated microprocessor-based computers. A database system should not be simply viewed as an application program of the mainframe. A database system should be viewed as a database server of the mainframe for on-line queries and high-rate transactions. As far as the mainframe is concerned, the mainframe operating system is the physical resource manager and controller and the backend database system is the logical resource manager and controller. In this way, the resources of the mainframe can be efficiently utilized and cost-effectively managed.

* The work reported herein is supported by grants from the Department of Defense STARS Program and conducted at the Laboratory for Database Systems Research, Naval Postgraduate School, Monterey, CA 93943.


26
3. THE DATABASE SYSTEM AS A LINGUIST

Conventional database systems are all mono-lingual and mono-modal. In other words, the database of a database system is viewed by way of a single data model and the data language of the database system is based on the model chosen. Thus, we have the relational database system with the relational databases and the relational-model-based SQL, the hierarchical database system with the hierarchical databases and the hierarchical-model-based DL/I, the network database system with the network databases and the network-model-based CODASYL-DML and the functional database system with the functional databases and the functional-model-based Deduplex. However, why should database systems be restricted to the single-model-and-single-model-based-language approach?

To answer this question, let us review the evolution of the operating system. In the 50's, the operating systems are of single languages. Thus, we have had for example, the FORTRAN monitor system and the COBOL monitor system. These monitor systems have had their own library routines, data sets, I/O routines, linkage conventions and other resources. It took another decade before the concept of common library subroutines, centralized file system, generalized I/O control systems and standard linkage conventions became popular. At the present, a modern operating system can support many programming languages and data structures. In addition to procedure-oriented languages such as FORTRAN and Pascal, a modern operating system has a centralized file system and supports logic programming languages, non-procedural programming languages and utility packages. Thus, the notion of programming languages, data structures of the programming languages and operating systems corresponds to the notion of data languages, data models of the data languages, and database systems. With such a correspondence, the evolution of the database system is likely to correspond to the evolution of the operating system. Modern database systems may therefore have common access methods, centralized storage allocation schemes, generalized disk I/O and standard language interfaces, in order to support multiple data models and their model-based data languages. The prototyped multi-lingual database system (MLDS) is an indication of what the future holds.

4. CONCLUDING REMARKS

In contrast to the operating system, the database system is the controller and manager of logical resources, namely, the database. As such, the database system and its database can best be facilitated in either the single-backend environment for stable and small databases or the multiple-backend environment for growing and larger databases. The use of backends allows the database-system software to be offloaded from the mainframe and to be away from the mainframe operating system. With dedicated backend hardware and specialized backend software, the performance of the database system improves considerably. By utilizing microprocessor-based hardware and specialized software, the cost of the backend system is more than compensated by the saving of retaining the mainframe longer, of the cost differential of the mainframe replacement, and of the freed-up physical resources due to the absence of the database system and its database in the mainframe. Database systems should not be treated as application programs by the operating system of the mainframe. Instead, the relationship of the database and operating systems is in specialization with the operating system on the physical resources of the mainframe and the database system on the logical and physical resources of the backends. In contrast, the mainframe operating system is clearly inadequate in supporting the modern database systems.

From an evolutionary viewpoint, the database system may be benefited by the progress made in the operating system. At the beginning of the evolution, all systems, whether they are operating or database systems, are mono-lingual. As the evolution proceeds, systems become multi-lingual. Although the operating system of the present day is highly multi-lingual, the database system has only begun to experiment with the multi-lingual functionality. This evolution from mono-lingual to multi-lingual in database systems seems inevitable. We should strive for multi-lingual database systems for the future.

ACKNOWLEDGEMENT

The author would like to thank Professor Jane Liu of the University of Illinois for including the author as a panelist in the panel on the relationship of the database and the operating systems, and for inviting the author to contribute this paper. Professor Liu has also played an important role as the editor of the first issue of the Data Engineering Bulletin, which included the proceedings of the first workshop on database-operating-system interface some ten years ago. The workshop was organized by this author with the help of a number of professors at Illinois, Ohio State, Purdue, Northwestern, and UCLA. It was attended by practitioners in operating and database systems. We have wondered why it has taken this long (some ten years) to have any forum on such an important and fruitful area.

REFERENCES