Preface

Today's computer environment is characterized by increased computer capacity and capability, by computing options ranging from total centralization to complete decentralization, and by advances in complex applications. This environment challenges the data processor to effectively and fully utilize available computing power. There is a need to study and plan data processing activity, considering such topics as performance tuning, capacity planning, distributed processing, and programming productivity. The papers in this issue explore some of these topics.

Much attention has been focused recently on the idea of distributing the data processing task among multiple computers and locations. Technological advances have made possible the cost-effective implementation of this concept, while the expanding role of the computer in every area of an organization has frequently made this approach to computing desirable. Within the scope of the distributed processing concept, there is a variety of definitions and implementations. The paper by Scherr introduces distributed processing and provides a synopsis of its characteristics, benefits, and potential concerns, such as data integrity and application growth.

Today's applications often involve access to and manipulation of very large data bases. Papers from England, by Gravina and Smedley, discuss the evaluation, design, and implementation of such applications. The National Westminster Bank of England has recently applied the IBM 3850 Mass Storage System to a large archival data base formerly kept on microfilm. This has allowed the on-line random retrieval of records for answering inquiries on account transactions up to three years old. Special indexing methods for file retrieval help to ensure satisfactory performance. Gravina's paper describes the application, the development of a solution involving the Mass Storage System, and the ensuing implementation.

The IBM United Kingdom Scientific Centre has been working with regional planners toward advancing the application of computers to physical planning. An objective was to develop an interactive data analysis tool for unstructured problem solving by professional planning people, who may not be familiar with computer operation. A prototype, the Urban Management System, was developed at the centre and is reported on by Smedley. Large, unformatted data bases were to be analyzed and operated on in unstructured ways. The requirements of the system were met by designing a user language that would be highly functional and easy to use.

The capacity of computers to handle large and complex applications has encouraged users to look for ways to install these applications quickly. As a result, program development method-

ology is receiving considerable attention. The use of engineering disciplines in program development is one way to enhance programming productivity. The Data Stream Linkage Mechanism (DSLM) is such a programming discipline; it emphasizes design modularity by the linking of precoded and custom-coded modules into a network through which data passes. Conventional programming techniques view programs as exact sequences of operations which match the serial operation of the computer more than the often asynchronous operation of the application. The DSLM approach views the program more as a data flow through work stations, as in a data processing "factory." This data flow aspect of DSLM, a prototype of which has been implemented in Canada, is the subject of the paper by Morrison.

Installing large applications requires an understanding of the amount of computer resource needed. There is a need to be able to predict and control the performance of a computer system. In the last issue of the *Journal*, a paper was presented that discussed performance tuning of MVS. The approach was that of "first-cut" tuning, in which operating system parameters were modified and results observed in a repetitive cycle until the desired performance was obtained. A more in-depth study of performance would normally involve the use of modeling. In this issue, two papers present models for capacity planning and performance tuning.

From Germany, Kraemer describes an analytical queuing model based on the DOS/VS supervisor. The model uses queuing methods in a set of APL functions that allow a flexible specification of the supervisor, configuration, and workload. In the paper by Chiu and Chow, an MVS performance model called PMOD, developed at the IBM Thomas J. Watson Research Center, is discussed. This model uses a combination of simulation, queuing theory, and statistics to ensure simple input and fast running times. The use of both models for capacity planning and systems tuning are covered in the papers.

Glenford Myers of the IBM Systems Research Institute in New York recently wrote two books, on program design and on computer architecture, which are reviewed in this issue. I would like to thank Luther Haibt and Burton Leavenworth for these reviews.

Connie Thiel
Editor