

Preface

The papers in this issue describe four separate topics: a relational data-base system, the interactive modeling of computer systems, a method of monitoring service levels provided by a computer center, and a technique for using microcode to execute APL statements on a System/370 computer.

The relational model has attracted a good deal of interest because it brings a new intellectual discipline to the field of data base architecture. Data base systems developed on hierarchical or network models have become widely used, but relational data bases exist primarily in test or prototype form. Todd describes one prototype system, the Peterlee Relational Test Vehicle (PRTV), which is expected to aid in solving some of the problems encountered in using relational systems. PRTV is a high-level system—one in which applications are expressed in a language meaningful to the user and in which data are represented and manipulated in a natural way. As such, it embodies research both in data language design and in efficient implementation techniques.

As computer systems have become more complex, the problem of constructing simple models to describe and predict their behavior has become more challenging, even as the need for these models has become pronounced. In recent years, some models have been developed and packaged for widespread use, in some cases as interactive tools for performance prediction. Although the models themselves have been quite simple, many of the tools have proved quite accurate, particularly those that accept trace data from existing systems as input. Reiser presents the potential user of these tools with a simple and straightforward introduction to the modeling techniques that lie behind them. In distinguishing between models and methods of solving models, he gives the reader an understanding of the differences between what are commonly referred to as simulation models and analytic models. Reiser describes a research tool developed to aid the solution of some simple models and presents examples illustrating the application of his methodology to computer performance problems.

Administrators of central computer facilities may encounter differences of opinion concerning the level of the services provided. The disagreements occur because "level of service" is subject to two definitions: one from the viewpoint of the provider of services and the other according to the user of these services. This problem can become acute when on-line service replaces batch computing, thus eliminating the personal interface between the computer center and its users. Lewis proposes *service levels* as solutions. The purposes of *service levels* are to establish between provider and user an agreed upon benchmark of expectation and a comparable measure of achieve-

ment, to provide a definable structure for evaluating user requirements in terms of the economic conditions and human factors that would justify a given level of service, and to provide a track record against which complaints can be objectively evaluated and acted upon. Based on three years of experience with service levels, Lewis develops an architecture and translates it into data processing terms by presenting the data base elements and data structure required for service levels.

High-level languages have been designed for the convenience of the programmer and must be translated into the language of the computer for execution. For most languages, this translation takes the form of a compilation, whereby the source language is compiled into the machine language. Another technique is to translate the source into some intermediate language which is executed interpretively. A third method would be to build a machine to directly execute either the high-level language or the intermediate form. Hassitt and Lyon describe a method of emulating an APL machine on a System/370. This technique makes use of the *APL assist*, a microprogram that enables APL expressions and defined functions to be executed at the hardware level. This paper describes the operation of the APL assist and its interaction with System/370 software.

This issue marks fifteen years of publication for the IBM Systems Journal. The Journal was started in 1962, by the IBM Systems Research Institute, to provide a channel for the exchange of systems knowledge, both theory and practice, within IBM. In 1965 the Journal moved to the Engineering, Programming, and Technology Staff at Corporate Headquarters, and, in 1972, it was reoriented toward those readers interested in enhancing the current and future practice of data processing. During this time, circulation grew to 55,000, with more than half of our current readers employed outside IBM. Particularly gratifying has been the Journal's acceptance by the business community, which has become increasingly sophisticated in its use of computers. Of our subscribers within the United States, 64 percent are employed in business, 18 percent in education, and 10 percent in government, with the remainder scattered. In addition, the Journal has attracted a solid international following, with more than 16,000 readers outside of the United States.

Since becoming Editor, it has been my practice to publicly thank those who have made significant contributions to the success of the Journal. The names of the Advisory Board, the Publications Staff, and the Editorial Staff have all appeared on the inside front cover. There are others, however, who, to my knowledge, have never been properly thanked for their contribution.

First are the many professionals, both inside and outside of IBM, who have, over the years, refereed papers for us. They deserve thanks, not only for advising us on the quality of our papers, but also for devoting time and energy to help authors strengthen their papers and make them more readable. Second are the members of IBM management who have encouraged their employees to write papers and who have supported the free flow of information between IBM and the larger technical community. Third, particular thanks to Homer Sarasohn, Director of Engineering Communications, who, for the last 11 years, has provided management guidance to the Journal staff and has supported us in our efforts.

George McQuilken
Editor