

Preface

Data storage systems are vital components of a computer complex. DASD storage control serves as a medium between the storage device and a computer system and has been largely transparent to users. The introduction of cache storage has increased users' awareness of storage control. In her paper, Grossman puts storage control as it relates to DASD and cache storage into historical perspective. She discusses the role of storage control in improved performance, function, and availability of DASD storage systems. The interaction of hardware, microcode, and software are reviewed in the evolution of storage control. These three critical functions have been performed in systems from the early days of computing up to the present time.

Distributed systems have become more prominent and commonplace with the increased use of computer networks. The paper by Summers is a historical treatment of the reasons for having distributed systems and some of the difficulties involved in their use. It focuses on software for systems that are connected by a high-speed local area network to form a local-area distributed system. Example network configurations are presented, including workstations, heterogeneous multiple computers, and file servers. The paper provides a viewpoint from which the value, function, attributes, and design of these systems are discussed.

Large networks that include systems of various sizes require newer techniques to deal with the large number of file movements requested by various applications. Many difficulties associated with older techniques used on small networks are addressed by Systems Network Architecture/File Services. Ashfield and Cybrynski describe how SNA/File Services fetches and stores files that are transferred from one system to another. They also describe other functions of SNA/File Services.

In addition, the growth of large SNA networks has created a need for the centralized management capabilities that are provided in Systems Network Ar-

chitecture/Management Services (SNA/MS). Change management is a concept that has gained in importance as networks have become larger and more complex. The change management capability is an enhancement to SNA/MS. Ballard, Farfara, and Heldke describe this capability, including the design selected and functions provided. They discuss the significance of change management and how SNA/MS uses SNA/File Services to accomplish its tasks.

Since its introduction, the REXX programming language has become a significant part of many programming environments. First implemented on systems running the Virtual Machine Facility/370 operating system, it is currently available on other operating systems, including Multiple Virtual Storage. Hoernes describes its implementation as a command language on Time Sharing Option/Extensions with all of the elements of a full-function language. He discusses some alternative designs as well as the final design that was implemented. Salient points of compatibility with IBM's Systems Application Architecture (SAA) are described, since REXX has been designated as the SAA procedure language.

As programs age, it becomes more and more difficult to correct programming errors and update the programs to meet current requirements. The original developers are often not available, and documentation may not be up-to-date or sufficiently thorough. Tools are being developed to assist programmers in analyzing old programs. With some of these tools, analysis is performed while the code is being read and run. Corbi presents an overview of work being done in this area in the IBM Research Division. He discusses the history and background of the effort leading to development of the tools, with details presented in the papers by Pazel and Cleveland that follow.

Maintaining programs incurs costs in both time and money. Ways to decrease such overhead are necessary in today's business environment. Pazel describes a tool called DS-Viewer that uses graphics to present

the data structures of a program and to show groups of data structures and their interrelationships in a program. The user has the flexibility of choosing data-item value formats or of choosing computer-assisted structures.

Cleveland discusses another tool that assists in the maintenance of software by providing an environment to aid programmers in gaining an understanding of the parts of a program. This tool similarly utilizes a graphics approach to present information about a program as multiple views, with the capability of exploring a particular view in depth.

A technical note by Agarwal, Gustavson, McComb, and Schmidt describes advances that are available in Release 3 of the Engineering and Scientific Subroutine Library. The advances are built on work that was presented in a paper by McComb and Schmidt, previously published in Volume 27, Number 4, of the *IBM Systems Journal*. Also discussed are some of the techniques used to optimize vector and parallel performance.

As the new Editor of the *IBM Systems Journal*, I am honored to be associated with a publication of such high quality and usefulness to its readers. We owe a debt of gratitude to the previous Editor, Gary M. Gershon, for his diligence in furthering both that quality and that strong connection to the readers. The *Journal* will continue to provide a forum for new developments in computer systems and programming, and continue to serve as a bridge between computer science and practical applications. To this end, it is important that our authors and readers are actively involved. I solicit your contributions of manuscripts, ask for your assistance in reviewing articles, and seek your comments and suggestions. Without authors, the *Journal* has no substance; without readers, it serves no purpose.

Gene Hoffnagle
Editor