Preface

This issue contains a collection of papers on five subjects: a process for defect prevention in software development, with experiences and examples; procedures and effective techniques for development of applications for the Operating System/2™ (OS/2™) Presentation Manager™; a methodology and tool support for requirements specifications using box structures; the moving of the Distributed Processing Programming Executive (DPPX) and customer applications from the IBM 8100 to the IBM ES/9370; and support for user interactions with systems through artificial intelligence.

The first subject is defect prevention. Mays, Jones, Holloway, and Studinski describe the Defect Prevention Process, its implementation, and the beneficial results. This process was developed and applied at the IBM Communications Programming Laboratory in Research Triangle Park, North Carolina. The authors show that defects can be prevented at a relatively low cost through causal analysis and process feedback. A significant side benefit of this approach is the evolution of a self-correcting mechanism within the existing process framework.

The second paper on the Defect Prevention Process describes its application and implementation at the IBM Myers Corners Laboratory in Poughkeepsie, New York. Gale, Tirso, and Burchfield describe that work and the resulting benefits, and discuss the flexibility and potential of this approach for the prevention of defects in software development settings generally.

Franklin and Peters present a paper on the second subject—the development of application programs for use under OS/2 Presentation Manager. The Presentation Manager provides a graphical interface with windows for applications, and this paper shows how that program environment can be used for effective development of new user applications. This approach results in consistent user interfaces, reusable user interface programs, and effective use of Presentation Manager capabilities.

The third subject for this issue is box structures as a methodology for requirements gathering and for information systems analysis and design in general. Odom describes the methodology and its use in the particular area of requirements gathering and specification, discussing the beneficial effect of such an approach on completeness, clarity, and traceability. An extended example of the use of box structures for workstation printer requirements illustrates the method and shows how it would be implemented in a practical situation. One important feature of this technique is the creation of usable requirements statements that lead the development effort and that can be effectively used and maintained as the system and its requirements evolve over time.

The second paper on box structures, by Tagg, presents the results of an effort to add this methodology and its associated graphics to an existing computer-aided software engineering (CASE) environment. This effort was accomplished through the use of a customizing tool in the CASE environment, thereby demonstrating this as an effective means for implementing new methodologies in existing environments. The resulting tools and environment are now under further study and practical use in IBM.

Moving the DPPX operating system from the IBM 8100 hardware environment to the IBM ES/9370 hardware environment is the fourth subject in this issue, and the topic of three papers. The first paper, by Abraham and Goodrich, gives the rationale for attempting the moving, or porting, of over one million lines of operating system code, which is generally the hardest code to move from one machine to another—especially since it is rarely designed with portability as a design objective. A feasibility study was performed, alternatives were developed, and the decision was made to port the code. More than

800 000 lines of code were successfully reused. All of this, the authors claim, shows the value of structured design and data encapsulation, both of which greatly aided the porting effort.

Goodrich and Loughlin describe the DPPX experience, focusing on migration of DPPX applications and databases from the IBM 8100 distributed processing environment to that of the IBM ES/9370. Because many customers had applications running on the distributed environment of DPPX with the 8100, it was vital to provide a straightforward and rapid means for migrating those applications, so that this significant customer investment would be protected. An important aspect of this migration approach was the coexistence of applications and databases on mixed networks of 8100 and ES/9370 hardware running the corresponding versions of DPPX. This paper provides insights into the process of developing the necessary migration plans. It also describes customer involvement in setting the strategy, the approach, and the priorities for migration.

The third paper on DPPX porting by Boehm, Palmiotti, and Zingaretti discusses the means by which the DPPX operating system was moved from the IBM 8100 to the IBM ES/9370. This effort required a new compiler for the new hardware, redesign of hardware-sensitive components, recompilation of other components, and extensive testing of the resultant operating system. Methods and tools were created to help develop, integrate, and test the new operating system on the ES/9370. Key among these were the staged approach, the use of specialized debugging tools, and the use of a simulated hardware environment prior to availability of the ES/9370 hardware.

Prager, Lamberti, Gardner, and Balzac present a paper on the fifth and final subject, the use of artificial intelligence techniques to support user interactions with systems. They have developed REASON (Real-time Explanation And SuggestiON), which offers suggestions and information when user interactions are incorrect and can be queried by users for assistance with the interface. REASON has been implemented in a multitasking environment with windows, but has been developed to be domain independent.

As the *Journal* begins its 29th year, it seems appropriate to state a few facts that sometimes escape us as we move from paper to paper. First, this publication is a quarterly refereed technical journal, which means that the integrity of each paper is ensured by a process that depends upon reviews of content, currency, and value by experts in the field. Second, it is intended for the software and systems professional community worldwide. So, the papers are written for a technically aware readership and are selected from submissions by knowledgeable authors around the globe. Third, the *Journal* has over 100 000 subscribers worldwide. Of those, approximately two-thirds are customers, technical professionals, and researchers; one-third are IBM employees; three-quarters are within the United States; and one-quarter are outside the U.S. We hope that authors, referees, and readers will continue to be active participants as we move forward into the 29th year.

The next issue of the *Journal* will be a special issue on the recently announced Application Development/Cycle (AD/Cycle™) architecture, direction, and tool strategy.

Gene F. Hoffnagle
Editor