Preface

Although computer system families have historically been well served by unique operating systems and user interfaces, the growing tendency to install personal systems to operate as terminals to, and in networks with, work-group systems and enterprise-level mainframes has led to fresh thinking on systems design and architecture. Interchanging people skills, programming, and data among systems requires both working standards and common supporting software for programmers to use in interacting with the user as well as the operating system. For IBM the solution was seen as not only providing these common standards, formats, protocols, and architectures, but doing so in a manner that allowed for future growth and extensions, and that allowed applications to exploit the unique features of specific systems. IBM has chosen to call this solution Systems Application Architecture (SAA).

This issue celebrates the emergence of what may come to represent a new generation of software: software offering significant enhancements in cross-system compatibility. SAA has moved quickly from concepts, to architectures, to implementations. With the announcement of the AS/400™ product family, major segments of these architectures are now available to users ranging from desktop through mid-size to large systems.

The paper by Wheeler and Ganek introduces the three elements of SAA. These have been termed Common User Access (CUA), Common Communications Support (CCS), and Common Programming Interface (CPI). These three elements supply a design for cross-system consistency across the PS/2®, AS/400, and System/370 product families. The authors indicate that not since the introduction of the System/360 has IBM made such significant announcements in programming systems. These new specifications are open and available to encourage customer and software-vendor participation in the development of applications that adhere to SAA.

Among the key objectives of SAA are providing support for distributed applications, including distributed processing and distributed data. The Common Communications Support of SAA provides facilities that are accessed using the Common Programming Interface and support protocols for intersystem communications. In his paper, Ahuja discusses CCS and its relationship to the other elements of SAA; he describes how CCS has been designed in accord with international standards, and additionally allows for connection to non-SAA systems through the open specification of the formats and protocols of its elements.

By using SAA's Common Programming Interface in designing applications that request data from network files and databases supported by SAA, the systems designer will have great flexibility to move the locations of both processing and data to meet evolving user needs.

Another major element of SAA is Common User Access, which is described in the papers by Berry and Uhlir. CUA supplies rules for designing the user interface across the three SAA operating environments. In doing so it employs contemporary user interface approaches by incorporating advances such as action bars, pull-downs, and pop-up windows. Emphasis in CUA has been extended beyond the considerations of ease-of-learning and ease-of-use to encompass a strong focus on the transfer of learning across different systems. Berry presents a number of the CUA design principles and strategies within SAA environments. Since CUA is a set of rules and guidelines, rather than a product, the paper describes the interface topics covered and the degrees of flexibility and choice afforded designers wishing to conform to SAA CUA.

The guidelines of CUA support state-of-the-art application development while allowing for subsets for use on systems and terminals unable to support the interactive dialog of personal computers. By allowing for change resulting from experience and new technologies, it provides for durability and extendibility over time. The paper by Uhlir further discusses the user interface, including the two levels of interaction

provided by the SAA Presentation Interface and the SAA Dialog Interface.

Just as CUA allows users to transfer their skills and learning gracefully and productively from one system to another, the Common Programming Interface supports the portability of applications and the transferability of programming skills and knowledge among different systems. It is through the CPI that programmers access the user interface, communications, and database elements of SAA. Wolford provides a brief introduction to the CPI in his paper.

Buchwald, Davison, and Stevens provide a definition of integrated applications as a goal for users desiring to share both local and remote data and use applications designed with consistent terminology and appearance. They assess the value of such applications in improving productivity and describe how SAA can help developers integrate applications.

In the paper by Dunfee, McGehe, Rauf, and Shipp, the authors reflect on their experience in designing a family of office systems applications using SAA. Their task spanned all four SAA environments—MVS, VM, OS/400™ and OS/2™. Furthermore, the application was designed to utilize cooperative processing and relational database access, and to be operable from programmable workstations and host display terminals, thereby utilizing virtually all the major elements and components of SAA.

The papers by Demers and Reinsch respectively cover distributed files and distributed databases in an SAA environment. Demers describes the continuing need for comprehensive distributed file support and differentiates file systems versus databases. File models of stream files and record files are defined, and levels of file distribution are discussed. The paper describes the role of Distributed Data Management (DDM) architecture within SAA in providing a common solution to distributed file processing.

SAA has adopted Structured Query Language (SQL) to provide the common user and programming interface for database management. The paper by Reinsch provides a look at the issues surrounding distributed databases by defining five distinct levels of distributed database support. As one steps through these levels, it becomes clearly evident how complicated the system design and support of a distributed database can be, and what considerations come into play from an application design viewpoint.

While centralized information processing continues to serve the needs of a majority of business applications most economically, there continue to exist a number of applications where the clustering of processing activity would lend itself to distributed processing. Implementing such systems, however, has not been an easy task—in large part because of the incompatibilities in database management, communications, and applications programming between the central site and the distributed systems used at remote locations. The paper by Scherr takes a new look at the centralized versus distributed processing question in light of the standardized interfaces and function provided under SAA. It would appear that many of the barriers to distributed processing have been taken down with the arrival of SAA. We look forward to innovative distributed application structures using the new facilities available with OS/2 and OS/400.

The final paper in this issue by Haynes, Dewell, and Herman describes the evolution and role within SAA of the Cross System Product application generator. Although customized coding will continue to be widely applied, there are a great number of standardized applications that are now becoming economically feasible across all of the computing platforms that are ideal fits for development using application-generator techniques. SAA now provides the facilities to support the migration of these applications from smaller prototype implementations to large-scale use—or the reverse, from centralized processing to staged distributed processing. The goal remains the same: Bring productivity efficiently and quickly to end users.

SAA is little more than a year old and users are now just getting the opportunity to work with its implementations in OS/2 Extended Edition and Application System/400. Just as OS/360 provided the basis for a multitude of successful mainframe applications, SAA through its compatible operating environments will also support a new generation of applications capable of running on mainframes, mid-size systems, and personal systems. Welcome aboard! It will be an exciting new journey for all of us in the systems industry.

Gary Gershon
Editor