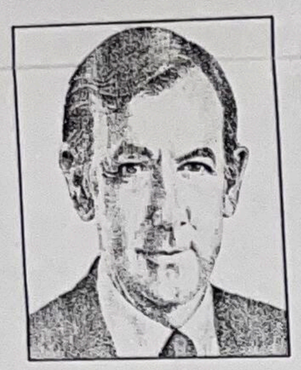
APPLIED INTELLIGENCE

SAA: Fulfilling the Need for an Integrated Environment



JAMES MARTIN

This is Part 1 of a series of articles on IBM's Systems Application Architecture (SAA). Integrated computing environments such as SAA will have a major impact on the software technologies of the '90s.

Systems Application Architecture, according to IBM Chairman John Akers, is the technology that will have the largest effect on IBM customers over the next five to 10 years. In many ways, the transition to SAA represents the most significant change in IBM software environments since the introduction of the System/360 in 1964.

As the name implies, SAA is an architecture for building enterprisewide systems of applications. But what exactly is a software architecture?

According to the dictionary, an architecture is "a style and method of design and construction." As it applies to software, the word has also come to mean a specification of the fundamental elements of structure. Thus, we can view SAA as the specification of both the fundamental elements of the enterprisewide application structure and the style and method by which those elements are combined to build applications.

SAA isn't a product or family of products for building enterprisewide applications. It's a consistent, open set of specifications for how these applications will be built.

Many vendors, including IBM, have recognized the need for integrated computing environments. Today, organizations often mix many types of computers. All the machines serve different, but sometimes overlapping, functions. They are programmed differently, have different user interfaces, and there's little communication between them. It's not easy to build applications that make the best use of differing machine capabilities.

The objective of integrated computing environments is to provide seamless connections between all the machines, so information and processes can be distributed freely among them. Additional objectives include the provision of consistent user interfaces for all applications. Such an environment provides programmers with a consistent development environment that makes it easy to build applications for this cooperative workplace.

SAA is the technology that will accomplish these objectives in the IBM environment. It's the architecture that will be used to build the cooperative processing applications of the 1990s.

IBM calls the computing system of the 1990s an "enterprise information system," or EIS. As shown in the graph, an EIS is typically composed of many different types of machines, including corporate mainframes, midrange machines and programmable workstations. The machines are seamlessly connected with applications working cooperatively across machine boundaries. Each of the machines is assigned the functions that it does best.

A primary strength of the programmable workstation is its ability to act as the user's window into the EIS. It provides a consistent and easy-to-use interface for all applications. The applications that are accessed might be executing locally on the programmable workstation, remotely on any other machine in the EIS, or cooperatively among multiple machines.

The corporate and midrange host computers are best at coordinating multiuser access to databases and transaction-processing applications. This host/programmable workstation specialization leads to what are called client-server applications. The host provides central services that are used by client applications on multiple programmable workstations (or other machines).

To work effectively in an EIS environment, applications must be designed and written to work cooperatively. The critical design function for cooperative processing is to divide applications into modules along boundaries that allow the functions to be distributed. For SAA, this involves isolating functions that support the user interface, database and program-to-program communication functions. Furthermore, the functional components of the application that are best suited for different machines must be isolated.

For example, an application that gathers data and updates a corporate database would divide its functions between the host and the programmable workstation. Support for the user interface, as well as the editing and validating of user input, would be performed on the programmable workstation in a highly interactive input session. Once the data is gathered and edited, the data would be sent transparently to the host for additional processing.

The host portion of the application would provide the database access and might also include more global validation of the data, such as ensuring that there are no conflicting entries in the database.

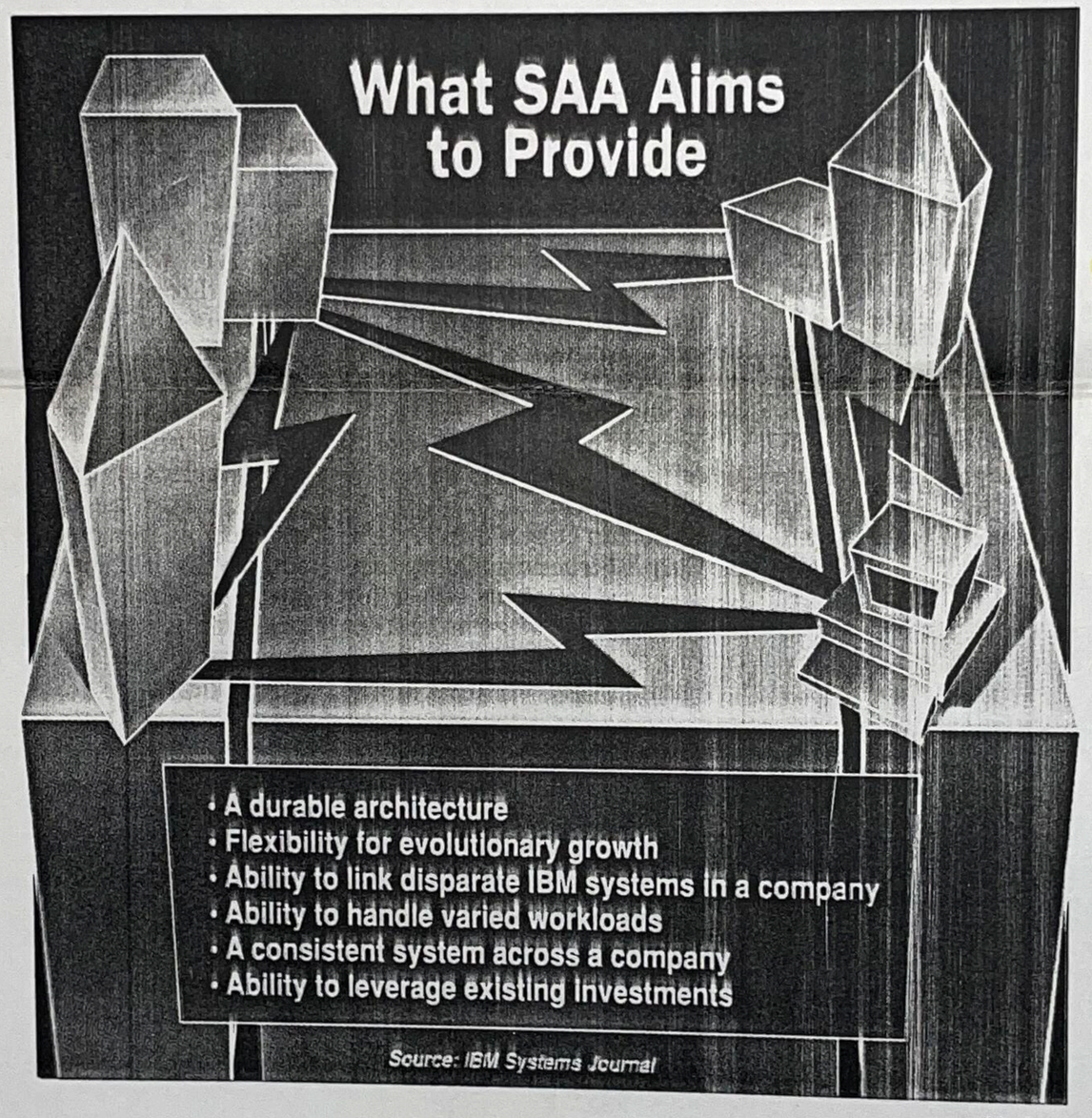
What exactly does SAA specify? Its primary function is to specify the three external interfaces to a computer. These are Common User Access (CUA), the user interface; Common Communication Services (CCS), the machine-machine interface; and Common Programming Interface (CPI), the programmer interface.

SAA-compliant user interfaces will give all applications the same look and feel. While primarily intended for the programmable workstation, CUA contains a subset that can be used for non-programmable terminals. This allows applications to take full advantage of menu-driven, windowing, graphical, programmable-workstation interface capabilities, and still maintain existing applications for non-programmable terminals.

CCS is the specification for interprogram communication. It is based heavily on IBM's System Network Architecture (SNA) but also includes support for X.25 protocols and Open Systems Interconnect (OSI) protocols. Most of CCS, such as LAN support, is shielded from the developer and user. Only the top-level program-to-program services are directly accessible.

CPI specifies the developer's tools, including programming languages and services. The products that implement the CPI will be used for building EIS applications. Different products will exist on different platforms, but to be SAA-compliant the programmer's interface to the products must follow the CPI specifications.

IBM has specified SAA for four main operating system environments: MVS and VM on the System/370, OS/400 on the AS/400, and OS/2 on the programmable workstation. The OSI and X.25 communication protocols allow connection to other non-supported environments.



John Avakian

SAA isn't a product for building enterprisewide applications. It's a consistent, open set of specs for how these applications will be built.

What About Unix?

While IBM would prefer that everyone use OS/2 on PS/2s, the company is aware that many users prefer Unix on workstations. For this reason, the development directions of IBM's version of Unix, AIX, parallel those of SAA.

The programming and communication interfaces will gradually become the same between SAA and AIX. This will allow Unix systems to be seamlessly incorporated as part of the enterprise information system. Organizations that prefer Unix will be able to make Unix, rather than OS/2, the window on the EIS.

The tools are not all in place yet for building EIS applications, but the architecture is well-understood. Organizations that wish to use SAA as early as possible should start designing and building new applications with the clear separation of functions required for SAA. As tools that implement SAA become available, the affected modules can be converted from current technologies to SAA technologies.

Next week, I'll describe how the architectures and standards embodied in SAA simplify the application-development process. ■

The James Martin Productivity Series, an information service updated quarterly, is available through High Productivity Software Inc., of Marblehead, Mass. (800) 242-1240. For information on seminars, please contact (in the United States and Canada) Technology Transfer Institute, 741 10th St., Santa Monica, Calif. 90402 (213) 394-8305. In Europe, contact Savant, 2 New St., Carnforth, Lancs., LA5 9BX United Kingdom (0524) 734 505.