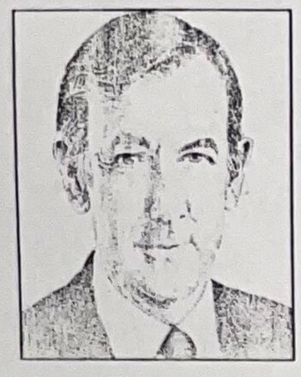
APPLIED INTELLIGENCE

SAA's Standards, Tools Promise Big Productivity Gains



JAMES MARTIN

This is Part 2 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.

SAA is based on a well-defined architecture and a set of consistent standards that define how applications are built. The use of standards and architectures is not new to the computer industry. Standards organizations have been in place for years.

There is always a conflict between standards-setting organizations and vendors developing proprietary products. A primary function of standards is to create an open environment. However, vendor-imposed standards are often designed to lock customers into a proprietary environment. Some de facto standards are set simply because one product becomes widely used.

The standards for databases, communication facilities and languages have all focused on the functionality supported by these technologies. Over the last few years the importance of standard application architectures has emerged. Many vendors are putting increasing emphasis on developing standard application environments.

Standards organizations are wrestling with the same issues. The Open Software Foundation was organized to address the issues of Unix environments and is using IBM's AIX as the basis for the future development of a Unix standard. Its goal is to provide an open framework for application development that does not lock a customer into a single-vendor environment.

The American National Standards Institute is also looking at these issues and is working on a standard for the critical base technology of a repository. A repository will contain full application specifications that can then be used to generate applications for particular environments. A common repository of design specifications is an essential component of computer-aided software engineering tools.

Users and vendors alike are realizing the value of specifying standard application-enabling architectures. Whenever a repetitive aspect of application development can be identified, a standard way of doing it can be specified, a tool to make it happen can be built, and the time it takes to build an application is reduced.

IBM's SAA is an impressive effort to unify IBM's application environments under a single architecture. The result will be tremendous productivity gains for users.

SAA provides standards for three interfaces to a machine: the Common User Access (CUA); the Common Programming Interface (CPI); and the Common Communication Services (CCS).

Let's look at how each of these encapsulates some of the repetitive work of application development.

Common User Access

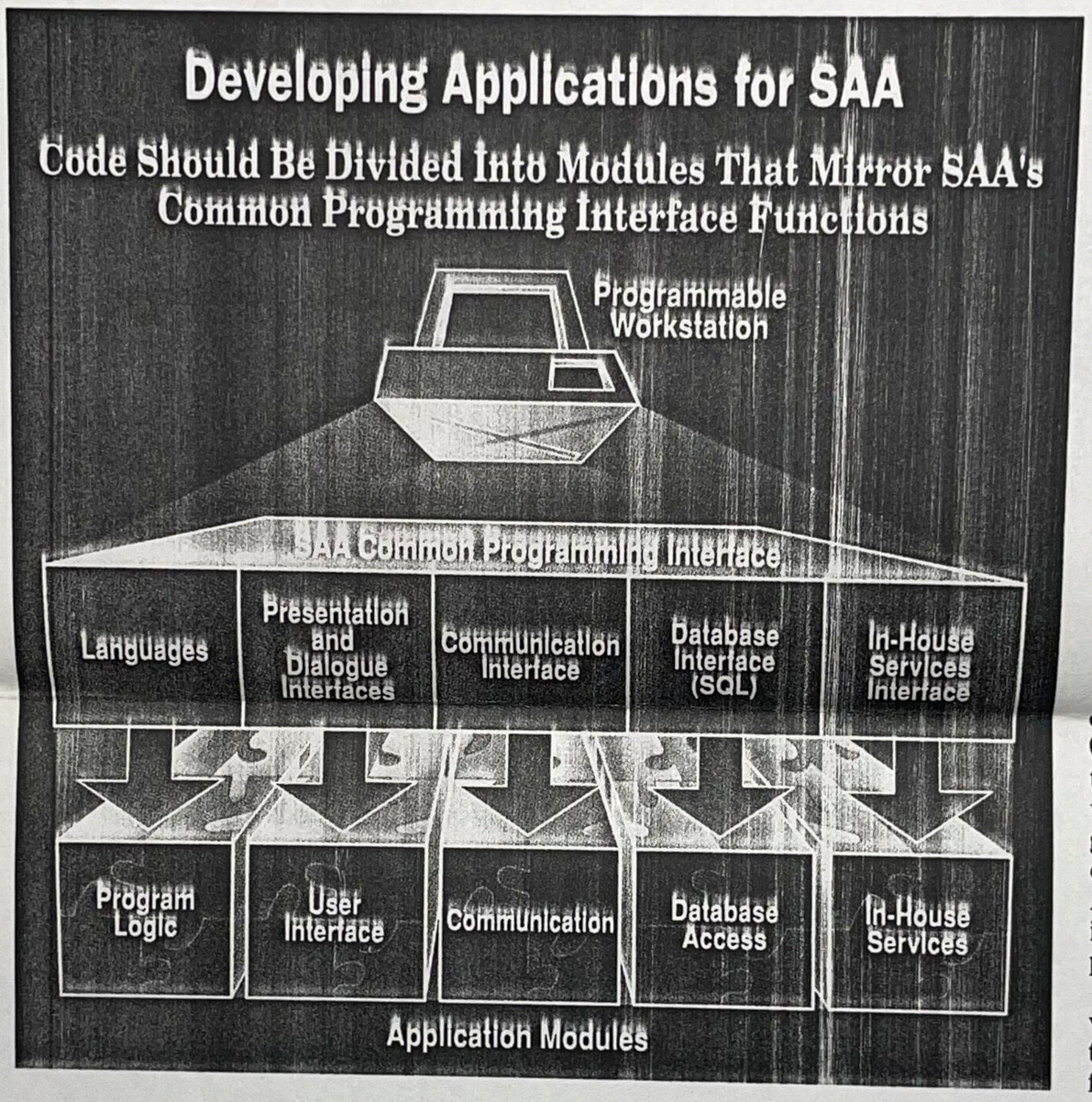
Every application has a user interface. The developer must design the user interface and implement it. In the past, user interfaces were developed at the convenience of the programmer. The usability of the interface was secondary to simply getting the application to work.

Today, everyone recognizes the value of a good user interface—it changes the relationship between users and applications. Poorly designed interfaces make the user a slave to the application and force the user to learn an alien syntax that is unique to the application. With a good interface, the user is in control and the application becomes a tool.

The problem is, it's difficult to design and build a good user interface. People disagree on what is the best interface. Without an integrated application environment, there are no consistent tools for building interfaces. An application that runs on one machine can't be ported to another because the tools for enabling the interface are totally different.

The CUA addresses both these issues. First, it specifies the design of the user interface, eliminating much of the debate over what the interface will look like. Then it specifies the tools that will be used to enable that design, thus providing for consistent tools across different hardware and software environments. In both cases, the architecture isolates and encapsulates a portion of the repetitive programming work.

The design chosen for the CUA is a good interface design, based on Xerox's research and popularized by the Macintosh. It was further refined and developed through user-interface studies at IBM, which enhanced it with technology acquired from firms with excellent user-interface reputations, such as Metaphor.



John Avakian

SAA provides a powerful means for programmers to avoid many repetitive and difficult tasks of application development not directly related to application-functional logic.

There are two levels of tools specified for enabling CUA interfaces. One is the presentation interface, which gives the programmer a high degree of control over building interactive applications, such as spreadsheets or graphics programs. The other is the dialogue interface, which allows the programmer to specify logically the components of the interface, such as menus and forms. These standard constructs do not have to be reimplemented, but simply logically specified. The tools do the rest.

Common Comm Services

The CCS provide the greatest architectural simplification for the programmer. The vast amounts of code needed to connect different machines is completely hidden. The programmer simply uses high-level verbs that specify program-to-program communications.

Again, the repetitive work of connecting machines is isolated and encapsulated by the SAA architecture. The architecture allows the programmer to concentrate on application logic rather than on communication implementation. Only the logical connections between programs need to be specified. Furthermore, these connections are specified in a consistent, standard fashion, independent of machine environment.

Common Programming Interface

The CPI contains language standards and service standards. The provision of common in-house services makes SAA a powerful standard. As shown in the graph, the language, user-interface, communications and in-house services are all specified as part of the CPI. The graph also illustrates that application modules developed using CPI services exactly mirror the architectural boundaries of CPI.

Another important service supplied by the CPI is database access. There are many different types of data- and file-management facilities available today, and each requires different implementations in different environments. The programmer has to decide which facility to use and learn how to use it.

With SAA, relational database management becomes the standard. Furthermore, the interface to relational databases—Structured Query Language (SQL)—is consistent from environment to environment. Because SQL deals only with the logical access of data, the programmer is freed from having to write code to a physical database design. Again, the data access and retrieval is isolated and encapsulated for enhanced programmer productivity.

SAA and the tools that enable it provide a powerful means for programmers to avoid many of the repetitive and difficult tasks of application development that are not directly related to application-functional logic. It does this by architecturally isolating certain elements of application design, specifying standards for those elements and providing tools that enable the standards.

Every organization using SAA will certainly benefit from SAA, but even larger benefits can be gained by extending the architecture to better model its own application-development process.

Next week, I'll describe the planning functions that are required today to prepare for the integrated computing environments of the 1990s. ■

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.