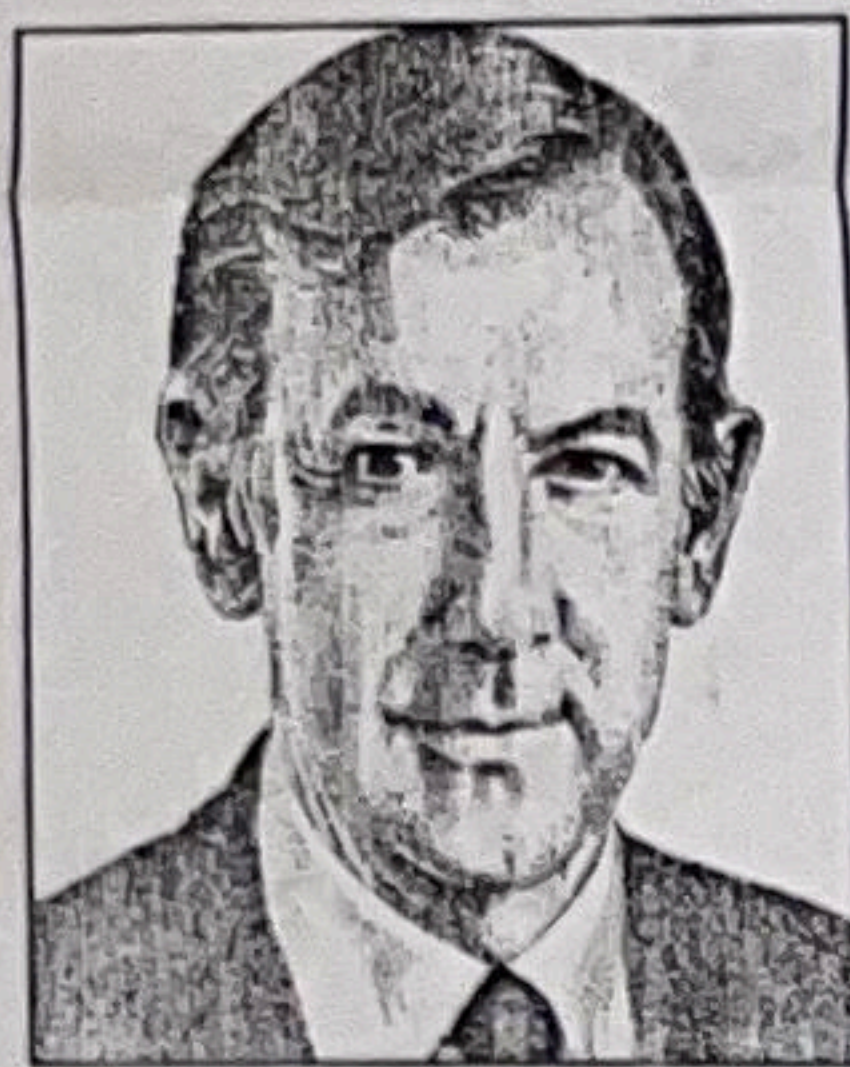**APPLIED INTELLIGENCE**

IBM's AD/Cycle Will Bring On a New Strategy for CASE

This is the third in a series of articles on the recently announced IBM Repository, part of IBM's applications-development strategy for the 1990s.

JAMES MARTIN

IBM's announcement of AD/Cycle is prompting many managers to reassess their applications-development strategies. But what is the significance of AD/Cycle, and how will it change the development process? Will it provide an open applications-development environment in which tools from multiple vendors can share design information in support of all phases of the development life cycle? What should a manager do to take maximum advantage of new development environments, methodologies and tools?

The details of the AD/Cycle announcement provide a clearer view of how applications will be developed in the 1990s in the IBM world. AD/Cycle provides the architectural framework required to build applications efficiently within Systems Application Architecture (SAA). It's based on a common repository used to store design specifications in a standard format.

Applications-development tools that comply with the standard Repository format can operate in an open environment and can share design information.

This has important implications for vendors of computer-aided software engineering (CASE) tools. CASE tools will no longer be based on proprietary, mutually incompatible repositories of design information. Instead, all compliant tools will follow a common definition of the objects used to specify an application; the tools will compete in offering users better ways to specify and manipulate these objects in support of the applications-development process.

The battle among vendors of applications-development products will be fought on the desktop, not in the mainframe arena. As shown in the figure, applications development will move more and more to the desktop environment, using integrated CASE tools that support many project analysts on a LAN. Executable code generated on the desktop will be exported to the target host computer. Several high-end, integrated CASE tools operate in this mode, generating code on the PS/2, which can be run on the PS/2 or exported to a specified host environment. Clearly, the desktop will be the dominant applications-development environment in the 1990s.

Contrary to speculation, AD/Cycle and SAA are not aimed at continuing the traditional dominance of mainframe computers. IBM is moving away from the conventional mainframe-centered software-development cycle toward a desktop-oriented development cycle.

SAA, as an architecture, supports the distributed client-server environment shown in the figure. In this environment, user-interactive processing is performed on the PS/2, and application-specific data is accessed through data servers distributed throughout the network.

AD/Cycle is ideally suited to support an applications-development environment based on the client-server model, which consists of PS/2-oriented CASE tools and distributed repositories of design information. Using CASE tools that are compliant with AD/Cycle, each member of a design team interacts with a personal repository used to store design information in a standard format. Specifications from analysts are consolidated in a departmental repository located within the LAN. Corporate-level

AD/Cycle Moves Application Development To the Desktop

PS/2s Generate Code; Hosts Act as File Servers

AS/400

S/370

Corporate Repository

PS/2 (External LAN)

Executable Code

File Server

Departmental Repository

Personal Repositories

John Avakian

CASE tools won't be based on proprietary repositories of design information. Compliant tools will follow a common definition of the objects used to specify an application.

---

data models and process models are typically accessed from a corporate repository on a host computer.

Development environments of the future, based on AD/Cycle and SAA, will relegate the mainframe to the role of a large file server, controlling and directing the flow of information from the corporate repository. Almost all of the functions of the applications-development life cycle will be performed on the desktop workstation, not on the mainframe. This mode of operation is used today in the operation of high-end, integrated CASE tools.

Therefore, a strategy for applications development in the 1990s should be based on identifying the best architectures, methodologies, tools and management techniques available. Strategic issues are important in defining how applications-development technologies like CASE will be implemented within an organization. A strategy for CASE must directly address critical success factors of the organization, such as the use of information for strategic advantage and the use of computerized systems to lock in customers and lock out competitors. An effective plan for CASE should be tied directly to the critical success factors for the business.

One of the most important critical success factors for many businesses today is support of business change. Everything in business is accelerating. The business metabolism is increasing and growing more efficient due to the widespread use of computers and the increasing speed of applications development. As the windows of opportunity shrink, organizations that take three years to build mission-critical applications are providing their competition with an opportunity to get into the market much faster.

In many cases, the inefficient applications-development methods used in information-systems (IS) departments are preventing the business from evolving. IS management must be able to use CASE technology and development environments, such as AD/Cycle, to support business change and ensure that the applications of the future will be able to evolve as quickly as the business requires.

One of the most important components of a CASE strategy is high-speed development, including the rapid development and maintenance of complex mission-critical systems that run major portions of the enterprise. The CASE strategy should be based on using the most powerful integrated CASE tools available and embedding those tools within a development methodology designed to build strategically important applications at high speed, in support of business change.

The most appropriate methodology for building strategic systems across a corporation is information engineering. The methodology was specifically designed to build detailed data and process models that can be converted automatically into code. Information engineering typically starts with high-level strategic planning issues and enterprise modeling. It moves through successive levels of analysis and design, building up sufficient information to generate code for an application automatically.
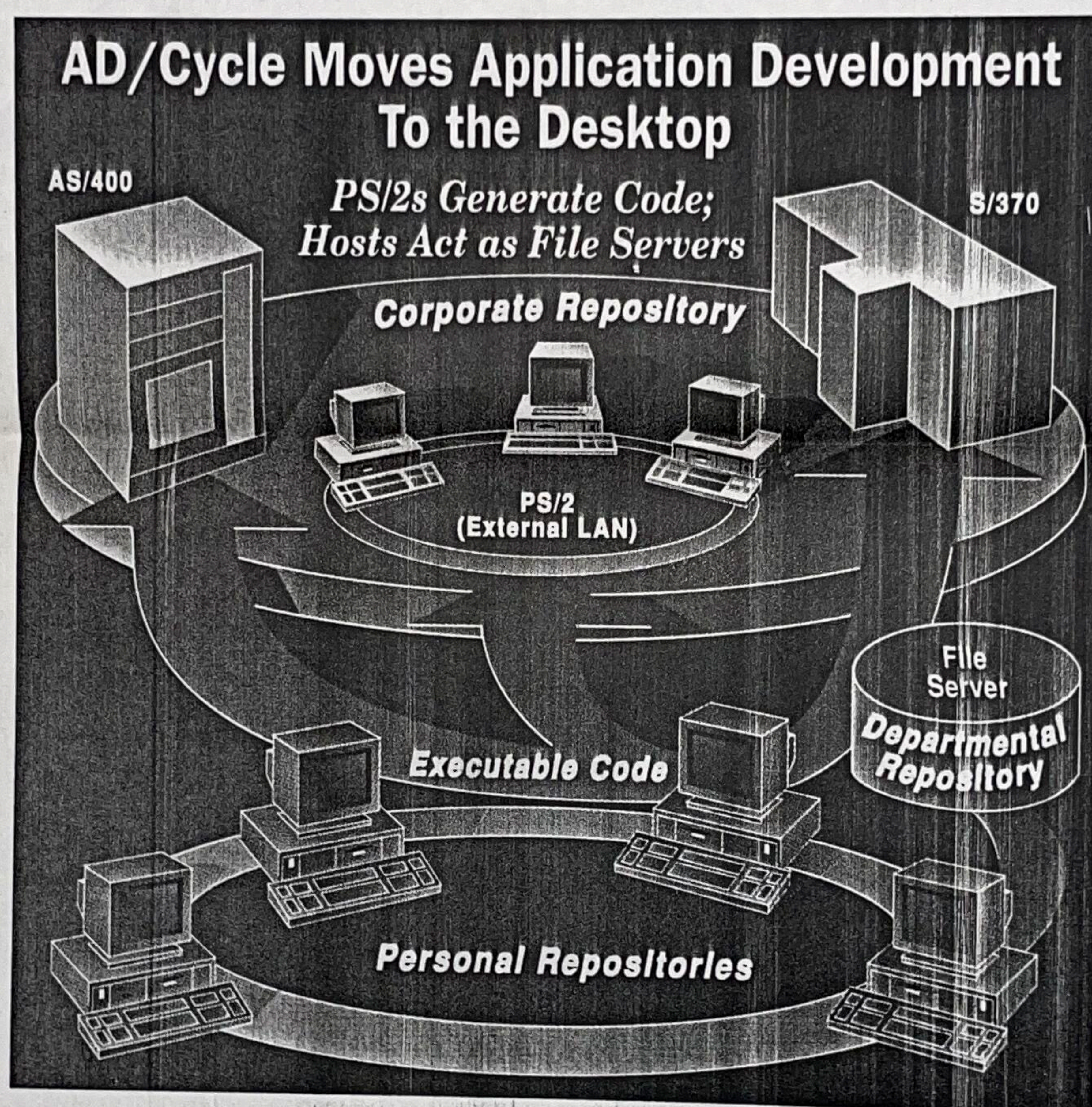
Recent extensions to the information-engineering methodology have provided support for high-speed applications development with small teams of specially trained analysts. A great deal of management attention is devoted to providing the teams with clear goals, a high level of support and maximum motivation for excellence.

Turning from Tradition

We're rapidly moving to a development environment in which businesses can no longer afford to use traditional structured development techniques. It's no longer possible to use separate teams to build separate projects that don't interconnect. Most competitive firms are now saying they must have application software connectivity throughout the value chain, which they cannot achieve without information engineering.

Information engineering is oriented toward support of disciplines such as strategic planning, enterprise modeling, complex data-modeling techniques, business area analysis, tight integration across projects and rapid applications development. If an organization is not using the advanced, highly integrated techniques embodied in information engineering, it is at a strategic disadvantage. Such techniques are of fundamental importance in developing a strategy for CASE.

Next week, I will discuss a strategy for introducing CASE tools that are compliant with SAA and AD/Cycle. ∎