REPRINTED FROM IBM SYSTEMS JOURNAL, VOL28, NO 3, 1989; © 1989, 1999

System overview of the Application System/400

by D. L. Schleicher
R. L. Taylor

This paper describes IBM's recently available general-purpose midrange computers—the Application System/400™, the basic intentions of the product, the significant factors setting forth system requirements, the primary design themes incorporated in the implementation of those requirements, and a description of some of the key system components. However, the paper is not intended to provide a complete system description.

The development effort for the Application System/400™ (AS/400™) family of computers began with an initial specification of requirements from which a set of system design factors was synthesized. These design factors did not map directly to the requirements but rather transcended them. Similarly, the design factors were synthesized into a set of system design themes which presented the overall approach for the implementation of the system. Finally, the system components were defined in terms of these themes. This paper is structured in the same way. It does not provide a complete description of the system. Such a description is available in "Advantage," *IBM Application System/400 Technology.*[1]

The AS/400 family is a new generation of general-purpose systems designed to address the total computing needs of line-of-business and office applications. It is a broad family of integrated systems offering considerably greater function and performance than its predecessor products, the System/36 and System/38, while maintaining accessibility to the large number of software packages available for those systems. It provides users with an integrated, state-of-the-art system that has a wide product range supported by a single integrated set of system software, improved usability for both new and experienced users, productivity tools for developing and using new applications, an application library of industry and cross-industry solutions, and on-line service and support facilities.

The fundamental AS/400 business strategy is to offer customers (i.e., users) business solutions for their data processing requirements. In most situations this strategy involves providing a ready-to-use integrated application set addressing the customers' specific needs. In other environments it is necessary to provide customers or IBM Business Partners (firms authorized by IBM that market, install, and/or maintain IBM products) with the ability to quickly and inexpensively develop new applications. From a product point of view, the strategy means that AS/400 offers applications, delivery systems, and support. Underlying this concept are the major AS/400 functional components: hardware, operating system and other system support software, and application development platforms.

This application orientation was the driving force behind the major design and implementation decisions. Examples of facilities intended to make appli-

 0018-8670/99/$5.00 © 1999 IBM

cation use and development highly productive include: conservation of existing application interfaces; integration in the system of data description and manipulation facilities to a relational database; consistent interfaces to all forms of work management including interactive, batch, office, and transaction processing; application development languages and utilities exploiting system functions; comprehensive communications features provided through device- and protocol-independent interfaces; query functions intended for general users; compatibility with IBM Personal Computers and Personal System/2® (PS/2®); and system management utilities supporting system operation.

System design factors

Complementing this overall objective of optimizing the support given to application solutions are four key influences which were the major considerations in the selection of the design approach and product content. They are:

1. Investment strategy
2. Customer requirements
3. Systems Application Architecture
4. New function (emphasizing additional advanced capability)

These factors, combined with the available hardware technologies, drove the decisions that led to the system structure of the AS/400.

Investment strategy. As in the development of any new system, classic tension existed between incremental enhancement and the implementation of new ideas. This tension presented potentially conflicting valid requirements. Although using the incorporation of new ideas as the fundamental genesis of design decisions may provide more effective solutions for new problems, the resulting inconsistency with previous offerings may preclude customer use of existing applications and skills. Thus, both new and incremental approaches affect the customer's costs. The AS/400 decision was driven by an assessment of the cost/benefit of each approach. For programming interfaces on which existing applications were dependent, interfaces of the most commonly used tools were provided, allowing portability of applications. Portability further implied that execution semantics be consistent while utilizing common system services across all interfaces. In contrast, for end-user display screen interfaces and for system operations, the training required to effectively use the new interfaces was demonstrated to be minimal, whereas the benefit of new design approaches was dramatic in terms of operational simplicity and productivity. Thus, new design technologies were employed.

Customer requirements. The primary requirements addressed by the system may be categorized into five areas: growth, simplicity, productivity, solutions, and support. The growth requirement encompassed three objectives: migration from a current system, which requires migration assistance tools as well as application portability while improving price/performance and maximum configurability; product range, which allows substantial growth in processing and storage capacity while avoiding any change in software design points or in operations; and "horizontal" growth through communications, which includes both homogeneous AS/400 networks and heterogeneous networks including other products such as System/370, PS/2, and System/3X.

The second requirement, simplicity, was evaluated from the viewpoint of each of the roles associated with the system (i.e., end user, system operator, application programmer, decision-maker, application vendor). Simplicity involves a number of topics including installation, use, training, documentation, operation, and customer support. The appropriate support level differs by the role and skill of the user, so in most instances both "usability" (simple and self-guiding for the inexperienced user) and "power" (productive and efficient for the professional data processing user) capabilities are provided with straightforward transitions between them.

The productivity requirement addresses the most expensive aspect in most installations: development of applications. Incorporation of programmer productivity features throughout the system allows more easily customized solutions to be available faster while reducing development cost, a benefit whether the application is created or purchased.

Customers are rarely interested in buying computers *per se*; they want solutions to problems. To meet this need, the fourth requirement provided comprehensive line-of-business software solutions for the special needs of users in individual industries, as well as cross-industry applications such as office software. Finally, allowing the customer to become more self-sufficient required system-integrated support functions spanning the range from hardware and software problem determination and service to marketing technical support and information through the AS/400 Electronic Customer Support (ECS) facility.

Systems Application Architecture. The AS/400 family fully supports IBM's Systems Application Architecture (SAA). SAA ensures the potential for future horizontal and vertical growth and enhances customer investments in application solutions, programmer education, and user training. SAA consists of four related elements establishing interfaces, conventions, and protocols (Figure 1). As one of these elements, Common User Access defines the design rules for user interface elements and interaction techniques. Common Programming Interface specifies the languages and services used to develop applications that are portable across SAA environments. Common Communications Support designates the communication architectures which, when implemented, allow the interconnection of SAA applications, systems, networks, and devices. The Applications element consists of Common Applications developed to execute across all SAA environments.[2]

AS/400 fully supports SAA.

As with all architectures, SAA specifies interfaces which, although independent of specific implementations, impose requirements on design. Satisfaction of the portability, compatibility, and interconnection objectives across the different SAA systems may be achieved through any of the following four approaches: commonality of implementation of the same hardware design optimized only to levels of price/performance; different hardware implementations of the same processor and I/O architectures, in each case executing the same operating system; common implementation of those high-level operating system components establishing the interfaces according to an architecture; and unique implementations of the specified interfaces.[3] By virtue of their decreasing levels of common implementations, these four approaches present increasing difficulty in achieving semantic architectural adherence. However, when new interfaces that provide function equivalent to existing capability are introduced, system resources must be shared efficiently between existing and new applications. Since system-specific function exists in the supporting implementation of the current interfaces, the best initial approach is to have system-specific implementations mapping the new interfaces to these existing functions. If the services are new, particularly if they employ SAA interfaces, the most cost-effective strategy is to have a common software implementation. Both of these approaches are employed in the SAA implementation on the AS/400 system. An example of the former is the Structured Query Language/400 (SQL/400) database implementation. An example of the latter is the execution environment for the Cross-System Product (CSP).

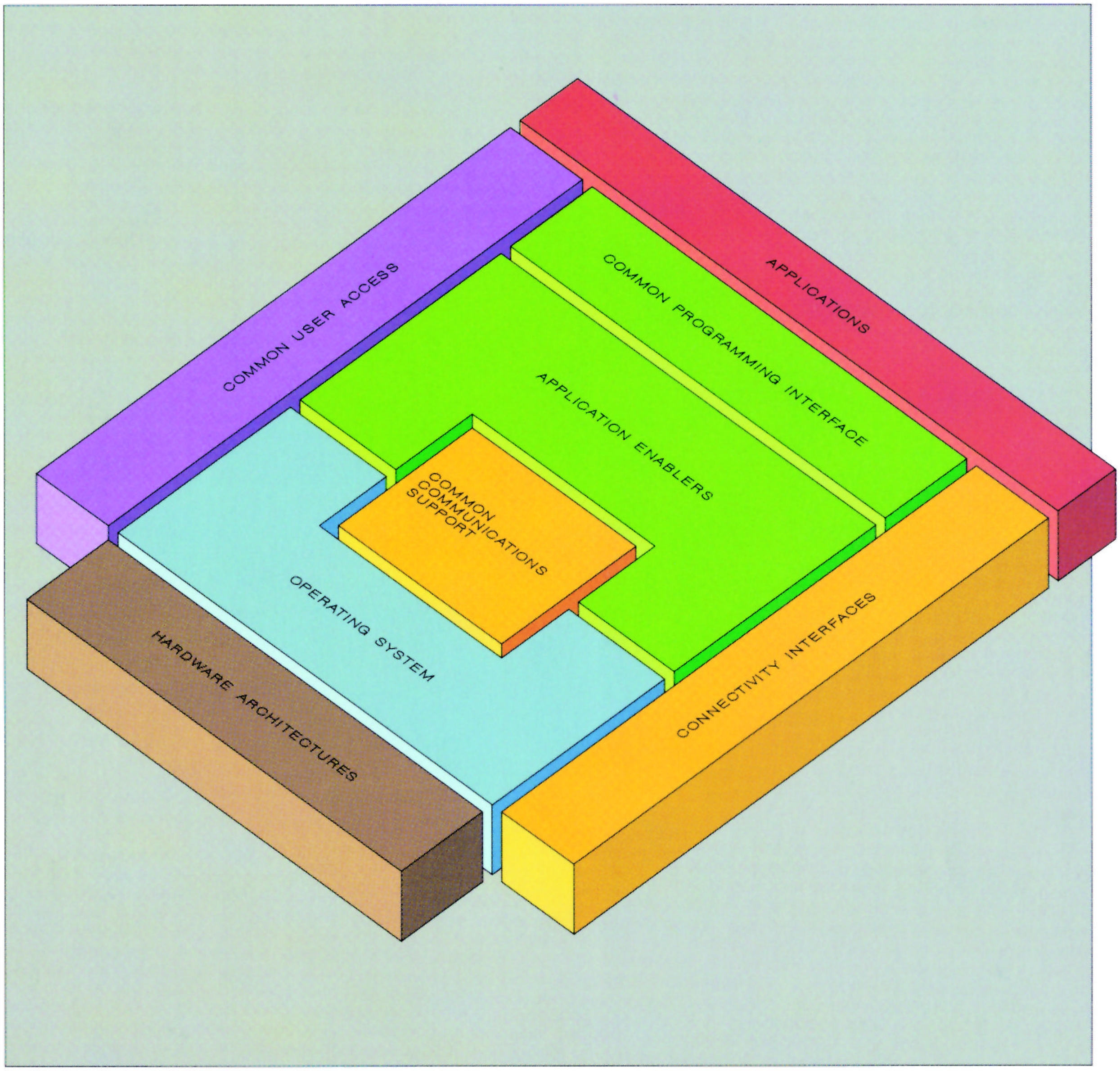
New function. Substantial new function is provided on the AS/400 system, much of which is not easily adaptable to existing interfaces and operations. Many areas in which new function is provided are described in the system description section of this paper. Primary objectives incorporating new function include integration within existing implementation layers avoiding subsystems, access via interfaces available at the appropriate point in the application, and implementation distribution across the system layers.

Design themes

In order to best address the requirements described above, the AS/400 family incorporates a number of advanced architectural and design approaches that characterize its implementation. It is structured in layers separated by opaque interfaces; it integrates system function through these layers to unify function; it allows the introduction of new technology by exploiting the opaqueness of the interfaces; it optimizes application execution for distributed processing; and it ensures the separation of program specification, data description, and system operation. Each of these themes is now considered.

Layered architecture. All systems may be characterized by their hardware and software components, the functional relationships between them, the control and data flow among them in delivering services to clients, and the way in which they are packaged. Typically, the packaging is described as a series of horizontal layers describing functional implementation, with the hardware at the lowest level and the software implementing the end-user interfaces at the highest level. The design of any external function

Figure 1 AS/400 SAA overview



proceeds through a series of steps iteratively, specifying implementation levels closer to the hardware. Selection of those interfaces that prohibit access in the execution of code (control flow) by supported layers (clients) of services other than those in the layer directly implementing the interface (implementors) permits multiple implementors, optimized to price and price/performance design points, while minimizing complex interactions in system execution and permitting the design to consist of independent, abstract components. Such interfaces are referred to as *opaque*. Thus, a system consisting of n layers, with the kth layer being opaque in order to prohibit such access, provides $n - k$ layers of access to an external function. The equivalent concept in the data flow is information hiding, in which data structures are created and manipulated without knowledge of how the implementation is represented. Inadvertent access is prohibited by virtue of having all code that supports functions performed on the data structure routed through code encapsulated with it. The extent to which functions and encapsulated data types are included in any particular interface designed according to an architecture is dependent on the confidence the designer has in the consistency of the resulting access limitations with the functional and performance requirements of the expected system use. Increasing the degree with which functions can be included in the opaque interface permits a design to capitalize on the de-

The system software is designed in layers.

creasing cost and improved performance of technology to implement function closer to the hardware. It can thus simplify systems programming.

Decisions on implementation layering for the AS/400 focus on these issues and provide the most significant structural characterization of the system (Figure 2). Specifying the required functional and performance characteristics for office and other commercial interactive applications, selecting the opaque interfaces wherein access would be limited, identifying the functions and data types to be structured at those interfaces, and measuring the system design against the specified criteria, all enabled the AS/400 system to be designed with significant levels of function at lower levels of system implementation. The AS/400 machine interface is the highest-level opaque interface on the system, requiring all supported function to be accessed through it by clients who utilize it. This high-level machine interface is made more functional by including: a generic instruction set for computation and control operations; systems services, communications support, database execution, and other more typical operating system function; specific office capabilities such as checking spelling; and an object orientation employing encapsulation to hide implementation techniques from the user and system software. These characteristics permit all software to enjoy the implementation simplicity, price/performance product range, and technology exploitation described above.

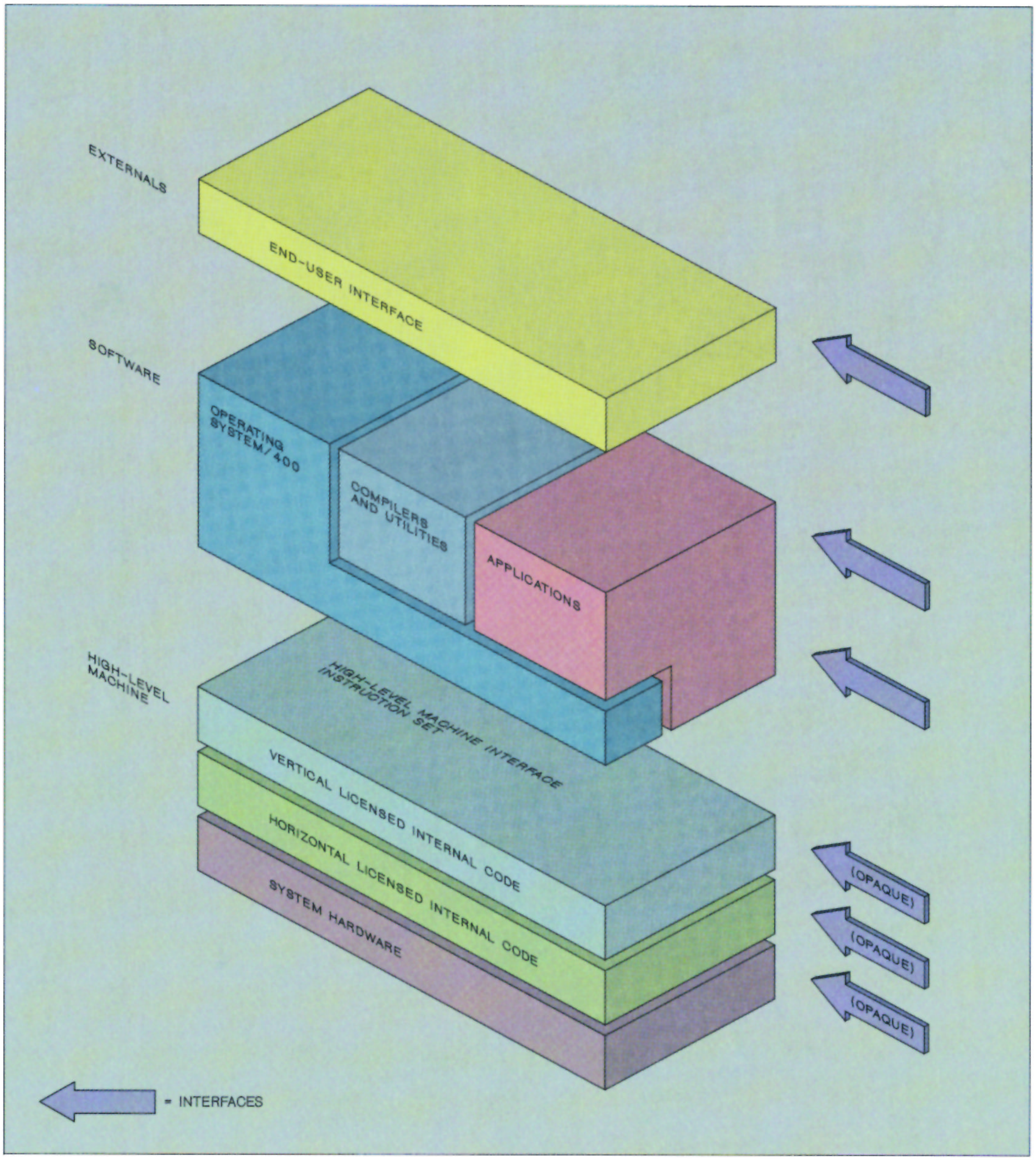
Supporting the opaque high-level machine interface are implementation layers, including another opaque interface, an execution interface which includes system supervisory and resource management function and one for the execution interface that supports device- and configuration-independent I/O attachment. The former permits implementation of the code supporting the machine interface without concern for the specific hardware implementation, whereas the latter permits implementation of the processor independent of I/O characteristics.

The system software is similarly designed in layers. The end-user interface provides menus, prompts, help facilities, etc. supported by the User Interface Manager (UIM) which enforces SAA interface standards. A control language provides consistent access to services for end users and application programs. A number of programming languages, utilities, and programmer services are available to develop application programs. Data-, file-, and screen-definition languages provide external interfaces for device, communication, and database file description. Interfaces allowing System/36 and System/38 applications to execute as though they were on these systems permit easy portability of application programs. An SQL query interface allows end-user access to the integrated relational database equivalent to utility or high-level language access. Although the implementation of all these facilities is portrayed as layers, interface access is available to all system software services.

The ability to support true data abstraction through information hiding is the most significant feature of the object orientation of the AS/400 family. An object is split into a specification part and an implementation part, with only the former seen by a client (Figure 3). Inadvertent or unauthorized modification of the object cannot occur, since the client can perform only the restricted operations permitted by the definition of the object. Implementation independence results from changes made by implementation programs that operate without client awareness. This decoupling of use from implementation allows problem repair or functional enhancement without any impact on the client.

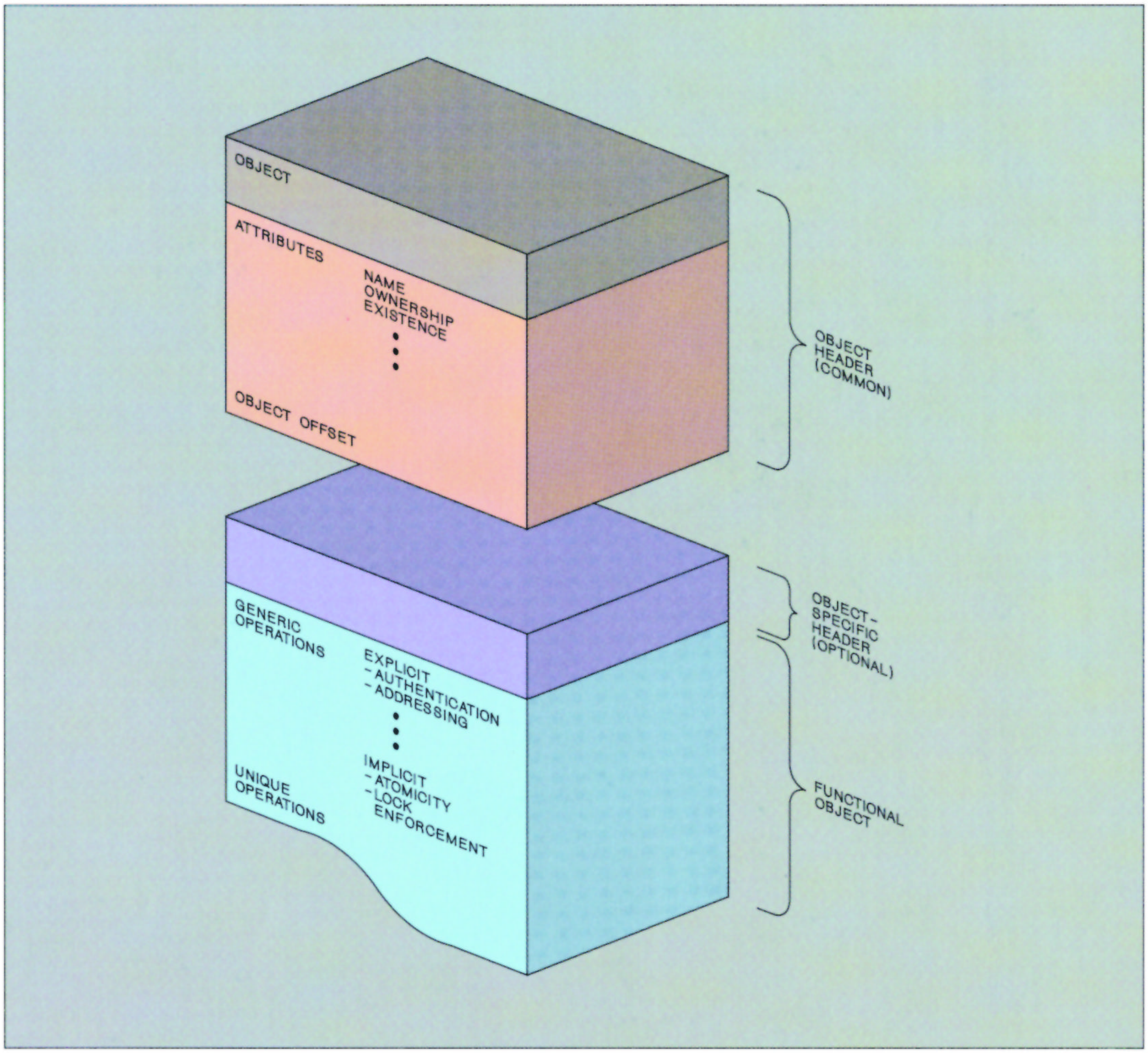
At the machine interface, objects are provided for all of the basic data types, and object management functions are supported, thus ensuring that objects

Figure 2 AS/400 layered architecture



are used correctly. The system software frequently combines several of these primary objects to form composite objects used by customer applications. Such combinations increase usability by allowing

Figure 3 Generic object structure



users to deal with very high-level constructs, improve reliability by reusing highly functional primitives, and allow performance optimization of frequently used functions.

Integrated system implementation. One implication of a layered system implementation employing opaque interfaces is the necessity of adding constructs consistently at each layer in the system when addressing significant functional enhancements not originally anticipated. To maintain the benefits of layering, these additions must conform to the concepts embodied in the interface architecture of each layer. As with the original implementation, the functional and performance characteristics of enhancements must be well understood to properly define and distribute the function. To the degree that software at the upper layers (including customer code) attempts to remedy any deficiency by implementing (at that layer) the required constructs, the more difficult it will be to ever effectively support that function. Separate implementations by multiple component developers (or customers) will inevitably result in incompatible interfaces, execution semantics, and system management functional replication. Not only does such action preclude eventual integrated system implementation without impacting the existing code, but it inhibits effective sharing of affected system resources.

The AS/400 object orientation provides substantial support for the integration of system function.

The advantages of integrating system function, as is done on the AS/400, are profound. Implementation redundancy is avoided, allowing for functional interface orthogonality. Resource management is consolidated, making system execution efficient and ensuring maximum sharing among system components and user programs. Services are available in a consistent, and therefore expected, way, allowing higher levels of productivity and usability. Communication among applications written in different languages and addressing different objectives can be easily facilitated. Requirements for having the user choose alternatives, based on function and performance, no longer have to be considered in application design. Choosing from a large variety of software packages performing similar, but not equivalent, function is unnecessary.

As a corollary to its layered structure, the AS/400 family of computers is designed as a total system, integrating all hardware and software components to avoid redundancy while exploiting the individual facilities of each member. This design theme is reflected throughout the system. For example, a user is given access to every function of the system and all of its data for which he or she is authorized with a single "log-on" procedure. The interfaces to system services are equivalent to application programs and end users for operational, programming, and maintenance functions via the control language. All services are available through menus, prompts, commands, and help information specified by the architecture. Context-sensitive on-line help is easily available to all users in all environments. A single program execution environment is provided for applications written in any language, allowing efficient program calls regardless of the language in which the code is written. A single set of resource management facilities is available for applications independent of type, whether they are batch, interactive, office, or transaction-processing. Office services are integrated into the system, permitting direct invocation by user applications. The relational database is integrated into the system so as to be directly available to all applications, thus avoiding the need to choose between flat files for performance and a database for functional capability.

The AS/400 object orientation provides substantial support for the integration of system function. For example, the specification portion (i.e., object header as seen in Figure 3) has a common part across all object types. It permits standard object-level functions, such as authorization management in support of system security, to be performed, allowing consistent operation across all object types. This in turn allows the external interfaces (e.g., high-level languages or control language) to provide consistent operation to end users.

Integration of AS/400 system facilities has three other significant attributes. The system as delivered to the user is ready to use when installed, though tailoring can be done, device configurations changed, and features added at any time. Subsystem implementation is avoided, allowing for consistent resource management, functional delivery, and external interfaces. Security levels allow the user to specify the degree to which and the mechanisms through which (i.e., specifying users authorized to an object or objects authorized to a user) security is managed.

The consistency promoted by the integration of system services yields a major benefit to the application developer. Since concurrent access to a wide variety of general-purpose applications which may be separately designed and implemented is an AS/400 user requirement, this consistency ensures that these different applications have the ability to effectively operate with each other, sharing system resources and data. This methodology promotes system code reuse and substantially reduces the costs of developing new applications.

Distributed processing. A key design requirement in the implementation of the AS/400 family is the dramatic change in the underlying architecture that is necessary to support the distribution of processing among multiple systems within a customer enterprise or among many different enterprises. Other requirements must be met in distributed processing. Including business and technical professionals as end users requires support for a wide variety of application and end-user needs. Since the customer's business depends on a network operation, integrity and availability of transmitted data are required. Having multiple media, data links, and networking architectures to choose from requires supporting multiple concurrent communications architectures which share physical resources, shield the application programmer from protocol detail, and allow communications with other IBM and non-IBM systems. Advances in hardware technologies that eliminate perceptible economy of scale advantages in systems encourage users to further distribute their processing in much smaller units, thus requiring support for large and dynamically changing configurations with ownership (i.e., control and scheduling of system resources) local to the system. Efficiency of operation within networks requires distribution of resources among different processors without the end user being aware of the physical location of the network and the physical medium.

Each of the above requirements is a continuation of classical trends in data communications. The AS/400 communications and networking architecture is structured for optimal accommodation of these requirements. However, in addition to the quantitative scaling up of communications requirements, there is a significant qualitative change to distributed processing. These two aspects result from two key technological advances. First, reduction in the cost and improvement in the performance of interconnection mechanisms allow the constitution of a system to be redefined. Second, the widespread and increasing use of programmable workstations (PWS) permits having highly interactive graphical user interfaces that utilize windowing, pointing devices, graphics, etc. This technology offers the opportunity to integrate AS/400 and programmable workstations, permitting the distribution of applications. The design of AS/400 capitalizes on this opportunity by providing cooperative processing. Through the use of a router running on each system, applications that employ multiple sessions between one or more PWSs and one or more AS/400 systems can be constructed. An example of such an application is provided by the AS/400 workstation function for office support, which allows the system to tailor the graphics data stream of the display. A similar, but more limited, capability is provided for host-dependent workstations in which text processing is distributed between the processor and the display I/O processor, which may be local to or remote from the AS/400 installation. The interpenetration of the AS/400 and the PWS represents a critical design factor for the current, and future, implementations of AS/400.

Independence of programs, data, and system. The requirement that AS/400 support applications written for multiple systems (i.e., System/36, System/38, SAA systems, and native AS/400) using multiple interfaces could have been addressed by building multiple bridges between the environments. To ensure the efficient sharing of resources, AS/400 provides multiple external views which are mapped functionally into a common underlying set of structures and functions. These views are designed to be complete, allowing all application programmers to map their use of system resources and services to a single system mechanism. For operations and end use, the system is designed so that users have a single, unified view of operational control, and a single end-user view through the standard Common User Access interfaces.

System functions are available through a unique layered structure with integrated system function.

This delineation of requirements by task required a careful categorization of the functions provided. The net result is a classification in which the objects created and/or manipulated by the end user, the environment established and managed by the system operator, and the applications written and enhanced by the programmer are individually defined and maintained. By providing dynamic binding mechanisms among the application programs, the objects on which they operate, and the system resources employed in their operation, the system permits mutual independence of each role, substantially improving the productivity and usability for each role performed on the system while providing efficient system operation.

System description

The AS/400 makes system functions available through a unique layered structure with integrated system function. Each layer provides a further abstraction of the data objects and facilities provided. A description of many of these facilities follows. For a more complete description of their implementation see References 1 and 4.

Hardware description. The AS/400 hardware structure is optimized to the high-level machine architecture, designed to initially give a wide range of configurability. There are two hardware packages. The smaller 9404 package utilizes a self-contained 0.65 m by 0.35 m by 0.75 m enclosure that includes the processor, storage, I/O controllers, magnetic media devices, power supply, battery power unit, and a single I/O bus providing a seven-socket logic enclosure with an expansion unit to the larger model. It is intended for use in an office environment. The larger 9406 package consists of models that can be installed in a 1.5 m rack in which logic cards are plugged into card slots on horizontal boards. Depending on the model, from one to seventeen I/O slots are available in its base configuration, with expandability provided through 12-slot I/O expansion card units connected by a cable of up to eight meters, thus permitting the I/O bus to span multiple racks.

AS/400 utilizes the densest standard cell and gate array chips ever used in an IBM processor. The system processor in the 9406 is implemented with a 2.5-micron oxide-isolated bipolar process with four levels of metal that permit over 14 000 equivalent two-input NAND gates per chip. The processor consists of six independent functional units implemented in six single-chip modules, each providing up to 240 functional I/O pins. Two of these units support main storage control, one performs virtual address translation, and three contain arithmetic logic unit (ALU) operations supporting computational and control operations. These six modules, with the memory for the control store and lookaside buffer, are packaged on a single processor card. This implementation achieves a 60-nanosecond cycle time to support the high-performance requirements of the 9406. Specifications for the technology are given in Table 1.

Table 1 AS/400 logic and array technologies

Logic	Bipolar	CMOS
Technology	2.5μ4LM	1.0μn-well DLM
Cells (wirable)	7,500 internal	27,720
Circuits (equiv. 2W)	14,000 (max)	40,000 (max)
Delay/Circuit (W C)	1.4 ns	2.1 ns
Power/Circuit	0.54 mW	0.1 mW
Power/Chip	7 watts	1.5 watts
Imbedded arrays	Personalized	Custom
Metal pitch	6.6 microns	3.3 microns
Size	7.4 mm sq.	9.4 mm sq.
I/O	240	231
RAM		
Chip bits	18K	144K
Module bits	72K	288K
Access time	20 ns	30 ns

Supporting the lower-cost objectives of the 9404 while maintaining the same logical internal code and software implementations required use of a lower-cost and easier-to-cool logic chip in the system processor. The same basic design is implemented in a 1.0-micron CMOS (complementary metal-oxide semiconductor) standard cell family providing lower power requirements, lower per-circuit cost, single-supply operation, and up to 40 000 equivalent gates per chip. The six system processor functional units are packaged into three single-chip modules, each having a maximum of 231 functional I/O pins. The system processor card contains these three modules, the memory for the control store and the lookaside buffer, four megabytes of main memory, and one I/O bus. This implementation achieves a 120-nanosecond cycle time while contributing to the low-cost requirements of the 9404.

As an objective, low cost was a major consideration in the 9404 package, but another important objective was modularity. The system unit is designed with individual removable modules for I/O devices, feature cards, feature power supply, and a battery power unit packaged so as to automatically connect to the appropriate power and signal connections during installation. These modules are contained in a metal enclosure designed to minimize electromagnetic radiation, both interference and discharge. The modular structure optimizes component cooling by packaging the fan as a part of the base power supply and including another fan with the tape and disk units. When upgrades are added, this building block approach provides flexibility with minimal changes to the system unit.

Computation and control. The processor does not directly execute program instructions. It converts them into horizontal licensed internal code control words which are directly decoded and executed. Thirteen different formats of 42-bit control words maximize hardware use through parallel processing of high-performance functions such as address computation and translation, status control and next control-word fetch, initiation of memory access, movement of data between processor registers, and a number of asynchronous operations. Control words supporting frequently used functions are contained in high-speed memory or control store of 4096 or 8192 locations, depending on the system model. Other functions have supporting control words contained in less expensive main memory. This structure allows a system to be optimized according to multiple price/performance models while permitting extendability in processor function without any impact on the processor hardware.

The horizontal licensed internal code supports the execution interface of the system which consists of a set of instructions similar to those on the System/370. Because this interface is not directly accessible by application programs, the implementation of individual instructions can be modified to reflect frequency of use in expected application environments. For example, the control instructions at the machine interface (MI) are high-level, composite, conditional compare-branch and test-branch instructions. The original implementation translated these control instructions into a series of executable instructions that provide this function. Performance analysis subsequently showed that frequency of use warranted direct implementation which permitted an isomorphic mapping from MI (see Figure 2) to the executable interface, thereby improving application execution performance. This example illustrates the ability afforded by the high-level opaque interface to radically change the implementation of system function without requiring modification, or even recompilation, of application programs, but allowing them to directly capitalize on the improved performance. As hardware technologies improve in cost and capacity, extensions to the executable interface can easily be made by moving functions to control words or even directly to hardware without concern for the impact on applications.

System services. The AS/400 machine interface is supported by vertical licensed internal code. Two categories of functions are supported within this code. As described above, the computational and control instructions are high-level, generic, composite instructions which map closely to those found in higher-level languages. These instructions are translated into executable instructions at the internal micro programming interface (IMPI) supported by the horizontal licensed internal code through an encapsulation process which creates an executable program object. A second category consists of a set of system functions managing critical system resources available through the machine interface supported by the vertical licensed internal code. These functions include storage management, database management, I/O processing, communications support, security and authorization, task management, and error recovery. Many of the unique features of AS/400 are provided by this layer. Extendability, as described for computational and control instructions above, is also characteristic of these functions in that technology improvements can be implemented without impacting applications. Reliability is improved because fewer instructions are necessary to perform a system function. Efficiency is enhanced because basic system resources are managed consistently across the entire system. Integrity is improved because the object orientation supported at this level provides capability addressing. Even the price of the system software is reduced because the consistent sharing of these facilities reduces implementation costs, permitting a lower price.

Memory management. A single-level storage implementation based on a single addressing scheme that handles all main and secondary storage is part of the memory management in the AS/400 family. At the machine interface, addressing is established through use of capabilities that allow repeated object access and is implemented through MI pointers that support 64-bit virtual addresses. When a program is encapsulated, these pointers are converted to 48-bit virtual addresses at the execution interface. (This provides yet another example of potential system expansion with application independence in that the system could expand addressability to 64 bits without an impact on applications.) MI pointers are objects that may be used only for addressing; they may not be examined or modified. As a guarantee, a tag bit is associated with each address that is maintained in main and disk storage with supporting processor logic to ensure address integrity.

The system employs an inverted storage directory; i.e., the page table contains virtual addresses rather than real addresses. Whenever any object is accessed, the user simply addresses that object (e.g., calls a program), and the system ensures that the object is loaded into main memory. This scheme permits the application to be independent of the stored location of the object and of the configuration of the magnetic media on the system. It allows an object (in particular, a database file) to be distributed across several spindles, permitting multiple extent objects to appear to the user as having one extent. The user is freed from concern with disk storage allocation since unused space can be managed by the system. This mechanism also improves data integrity since only one copy of an object is used at any time. Management overhead is reduced and synchronization problems endemic with multiple copies are eliminated.

A fundamental problem in multiuser system design is the difference in memory and I/O performance.

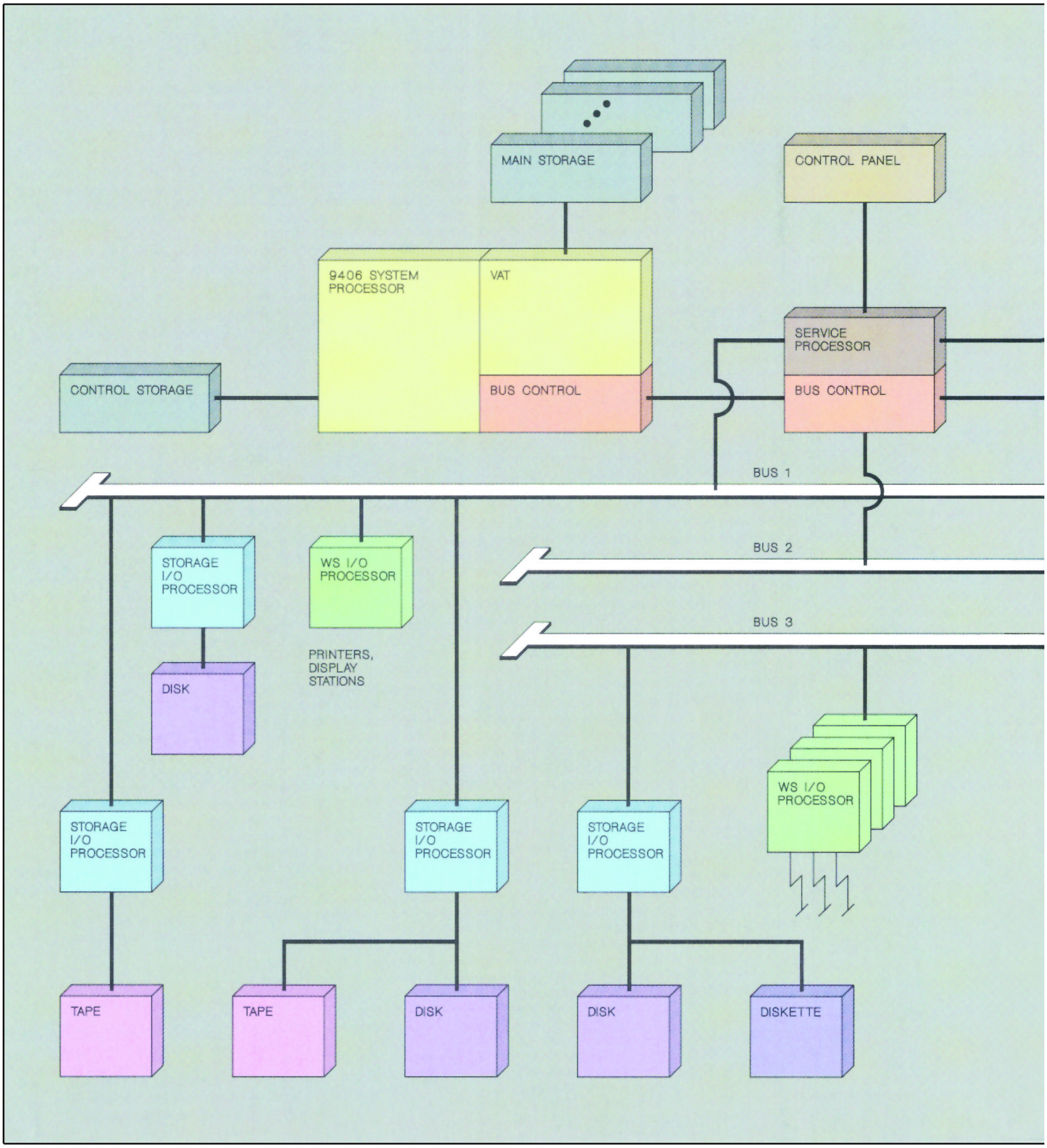
Input/output management. One of the fundamental problems in multiuser system design is the difference in memory and I/O performance. System performance depends on ensuring the rapid transmission of data between the system processor and I/O devices. AS/400 solves this problem by buffering I/O operations in memory allocated to each class of I/O activity. Control is provided by I/O processors specifically made for these I/O classes. This arrangement further permits offloading the system processor by using these I/O processors to handle compute-intensive operations such as I/O protocol handling, device error recovery, data link control for communications, or keystroke processing for workstations. Distribution of function between the system and I/O processors is mapped into the appropriate architectural structure (e.g., Systems Network Architecture [SNA] or Open Systems Interconnection [OSI] communications, disk or tape magnetic media), allowing physical devices to be shared while employing different higher-layer architectures.

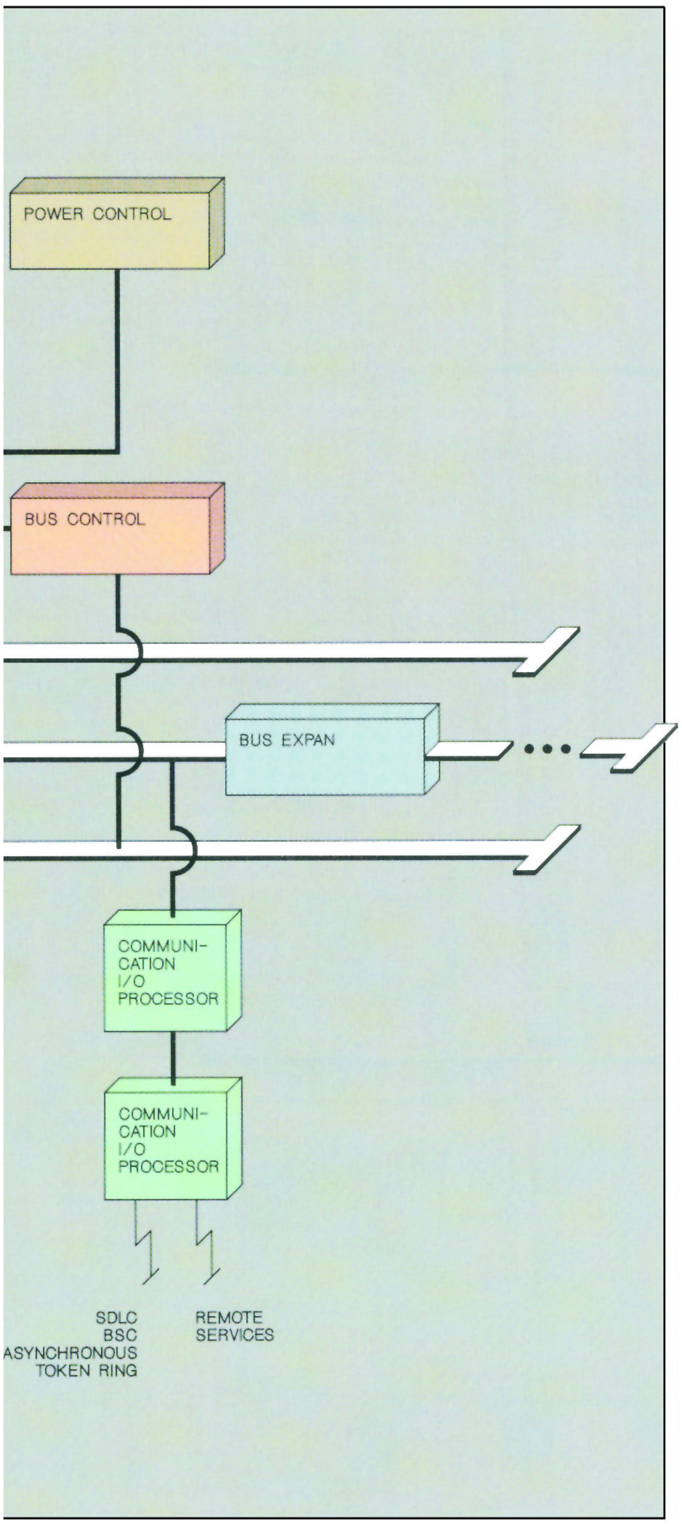
The AS/400 I/O hardware consists of the bus control unit (BCU) contained within the system processor; a model-dependent number of I/O buses and bus units (designed to provide a total of one BCU and 31 independent I/O units); I/O controllers for magnetic media, communications, and workstations; and up to three bus extension units for each bus (Figure 4). Linking the system and I/O processors is a 32-bit bus that provides full direct memory access (DMA) with queued, fixed-length messages sent in variable-length packets. Configuration extendability and flexibility that allow hardware repackaging, transport mechanism independence, and function distribution are ensured through logical addressing, location-independent service, a bus manager controlling the flow of data and control information across the bus, and by resetting the configuration at each initial program load (IPL), using vital product data (nonvolatile information kept in each I/O bus unit). Fault detection and isolation are handled by means of error-capture techniques, allowing failure recording as it occurs. This method improves intermittent and permanent fault analysis.

Each I/O processor is supported by unique multitasking operating systems that allow multiple concurrent operations. Since the system recognizes I/O configurations at IPL time, the appropriate control code is loaded into the I/O processor memory. For communications, recognition permits initialization for specific protocol tasks. The control code of the magnetic storage controller maximizes the device transfer rate while permitting multiple concurrent operations. The workstation I/O processor supports not only the device control functions but also end-user application functions. These functions include field editing, keystroke processing, text-processing, and national language support. The controllers may be locally bus-attached or remotely communications-attached and transparent to the end-user application. They may support a wide range of devices with common application and operating system interfaces by means of functional layering of the control code into components consisting of support for central processor attachment, device attachment, and common function components.

Personal computer support. A critical AS/400 requirement stems from the need for personal computer users to integrate their IBM PC and PS/2 usage. The availability of high-performance, programmable workstations with improvements in terms of function, overall system performance, integrity, data sharing and security, and the application extension

Figure 4 AS/400 hardware structure





available with an AS/400 departmental system that permits customers to capitalize on their overall data processing investments, required the efficient integration of personal computer support. Making this requirement prominent is the knowledge that significant advances in interconnection technology are occurring, permitting the performance-efficient distribution of application function to such high-performance programmable workstations.

AS/400 addresses this requirement through a set of facilities that allow the user to interact with the personal computer attached to the system as if he or she were dealing with a single system. The PC Support Organizer menu allows the user to select an application without knowledge as to where it will execute. When an AS/400 request is made, a router selects the appropriate connectivity option, permitting attachment independence. Up to five active sessions are available on the personal computer, permitting interaction with multiple AS/400 systems, with multiple tasks on a single AS/400 system, or with a combination of both. This capability can support remote AS/400 operations and network management, as well as application execution. The primary requirements of data sharing, integrity, and security are provided through AS/400 shared folders. By mapping personal computer functions to AS/400 functions while supporting naming conventions for personal computers, all transparent to the user, the full capability of the advanced, database system is available to that user. Files may be transferred to and from AS/400s and personal computers, with the appropriate data conversions automatically performed. AS/400 printers may be used as though they are attached to a personal computer. The distributed data management (DDM) capability can be used to access data from any AS/400 system in the network to which a specific personal computer is attached.

Electronic customer support. A key AS/400 objective is to maximize customer system and application availability. It is partially accomplished by improving the self-sufficiency of the customer through integration of Operating System/400™ (OS/400™) facilities that support hardware service, software problem detection and analysis, IBM service support access at time of failure, access to IBM marketing information, and on-line system technical support. By initiating a task from an AS/400 workstation, where the task is running concurrently with customer programs and using a standard communications interface to IBM marketing and service support systems, customers can dramatically improve their productivity whenever IBM support is required.

The necessity of having remote service support imposed critical hardware and software design requirements. As described in the section on I/O management, vital product data are kept in nonvolatile storage. In addition to being used for automatic system configuration at IPL, the storage is used for ordering, service verifying and activity recording, locating failing units, and error recording. These functions are supported by the implementation of a common architecture that provides consistent interfaces from all of the I/O processors. This consistency allows device-independent support in the operating system. Problem management, consisting of automated problem analysis, logging and tracking, reporting, and correction, is provided not only for system-detected hardware and software problems, but also for user-defined problems. Remote problem determination is significantly enhanced through the use of a copy screen image function that permits multiple viewers to concurrently access the same display.

Remote service support imposed critical hardware and software design requirements.

Electronic customer support functions for marketing support are also integrated into OS/400. They include a question-and-answer facility, access to IBM marketing information, and a technical information interchange facility. The first facility contains answers to frequently asked questions. This facility is initialized with IBM-supplied information and is expandable by customers or third-party application providers to support customer-specific applications. In addition to the database provided on the customer's system, remote databases can be provided. IBM permits access to databases supporting questions and answers for its service and marketing organizations, and, again, remote databases may be provided by the customer or a third party. A second facility allows access to marketing support tools such as announcement material and the AS/400 configurator. The intent is to provide the customer with the same information used to support the IBM sales organization. The technical information exchange (TIE) provides an electronic mail box for customers, allowing files to be sent to or from marketing support, thus permitting the most efficient exchange of information. These facilities are all designed to work together to enable users to manage their AS/400 installation more efficiently.

Database. Implementation of the OS/400 database support exemplifies two of the major design themes specified above: functional integration and multiple interfaces by a single set of system constructs. The fundamental distinction between the database provided by the AS/400 family and those on other commercially available systems is the level of function provided within the base system. The latter typically support file management, permitting physical record access with user-specified access paths to data sequentially, by index, or randomly through hash keys; access protection on a file basis; and limited recovery support. Semantic interpretation and management is the responsibility of the application program. Database management systems are then implemented independently of the system hardware, control code, and operating system, and run as subsystems presenting their own interfaces and managing their own resources.

In contrast, the integrated AS/400 database management provides capabilities to: define and manipulate data; create logical files which may map data from a single physical file, join fields from multiple physical files, access several physical files with multiple format definitions, or define a single record definition from multiple physical and other logical files; manage the physical and logical files including ownership, authorization, and specification of relationships for file cross-referencing; semantic interpretation of externally or dictionary-described files that allows enforcement or conversion of data types; define a transaction boundary to permit commitment or abortion of multiple related database operations; and journal records to permit file recovery, audit trails, and job accounting. Integration of the database support provides consistent syntactic control and definition facilities with other components of OS/400 for such generic operations as moving, accessing, authorizing, saving, etc. It also permits system management of all system resources, including database requirements. An example is provided by the consistent virtual memory management for system and database paging operations, avoiding conflicts and extra I/O operations.

The requirement that AS/400 execute applications originally prepared for the System/36, System/38, SQL, or native AS/400 environments results in multiple interfaces mapped to an integrated set of system functions. Not all of the AS/400 facilities are available to all of the interfaces since their original system implementation did not include them, but those that are common have consistent semantics. This consistency permits users of one environment to process files created with the use of other interfaces. This allows coexistence of all the environments, the staged conversion of old applications to capitalize on new AS/400 features, and use of improved data definition and query facilities for all data, both that associated with new and that associated with old applications.

Summary

The AS/400 system is a state-of-the-art, midrange, application-solution-oriented system employing a number of industry-unique architectural characteristics such as a formal layered architecture, system object orientation, integrated application and data facilities, independence at the system level of programs, data, and system, and a single-level storage implementation of memory management. It features the most recent hardware technologies supporting highly competitive price/performance implementations, advanced database capabilities, integrated customer support functions, and distributed processing capabilities, including the integration of personal computers. It is designed to fully support IBM's Systems Application Architecture. While protecting System/36 and System/38 customer investments, it provides architectural extendability to capitalize on future technology improvements and application requirements.

Cited references

1. "Advantage," *IBM Application System/400 Technology*, SA21-9540, IBM Corporation (1988); available through IBM branch offices.
2. E. F. Wheeler and A. G. Ganek, "Introduction to Systems Application Architecture," *IBM Systems Journal* **27**, No. 3, 250–263 (1988).
3. H. Lorin, "Systems architecture in transition—An overview," *IBM Systems Journal* **25**, Nos. 3/4, 256–273 (1986).
4. B. E. Clark and M. J. Corrigan, "Application System/400 performance characteristics," *IBM Systems Journal* **28**, No. 3, 407–423 (1989, this issue).

David L. Schleicher *IBM Programming Systems, 2000 Purchase Street, Purchase, New York 10577.* Mr. Schleicher is the systems director of Systems Application Architecture. He joined IBM as a junior programmer in 1964 in Omaha, Nebraska. His first assignment was in the former Federal Systems Division, first in Omaha, then in New Jersey. In 1971 he transferred to the IBM facility in Rochester, Minnesota, where he was initially responsible for managing internal IBM support systems and new products. In 1974 he was appointed manager responsible for developing major Verticle Microcode components for the System/38. Mr. Schleicher held a number of System/38 programming management positions prior to being named systems manager of Advanced Systems in 1983 responsible for directing all activities associated with the development of the System/38. In 1984 he was appointed Rochester Programming Center manager responsible for development and support of all System/3X and AS/400 programming. He was named director of the Rochester Programming Center in 1986. He assumed his current position in July 1988. In this position he is responsible for providing technical leadership and coordination for all IBM locations involved in the definition and implementation of Systems Application Architecture.

Roger L. Taylor *IBM Application Business Systems, Highway 52 & NW 37th Street, Rochester, Minnesota 55901.* Mr. Taylor is the manager of AS/400 Programming Work Bench. He joined IBM as a junior programmer in 1964 in Rochester. His first assignment involved development and support of 1401/1440/1460 software. Throughout his career, he has been involved in technical, management, and staff assignments associated with the IBM Rochester product line in programming, architecture, engineering, and planning organizations. In his current position, Mr. Taylor is responsible for managing internal IBM support systems related to AS/400 and other system programming development activity.