Common Communications Support in Systems Application Architecture

by V. Ahuja

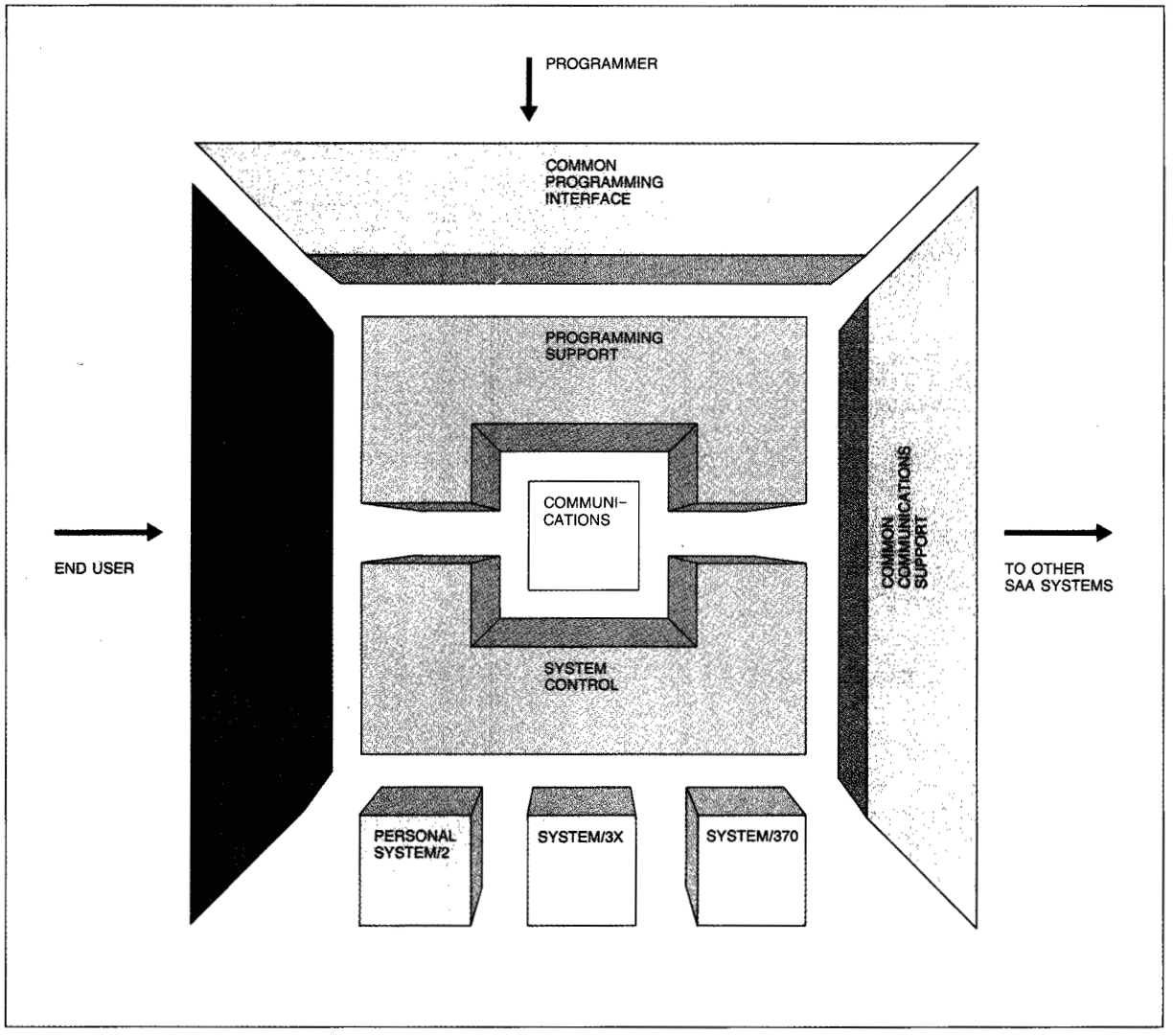
Application execution in a Systems Application Architecture (SAA) network depends on the underlying capability of the network to obtain reliable connectivity and orderly data exchange among its system components. The objectives of SAA are distributed applications, distributed processing, and distributed data, which are achieved through interconnected SAA systems supporting appropriate interfaces and architectures. The Common Communications Support of SAA affords this capability by utilizing a number of Systems Network Architecture communication architectures and international standards. These architectures provide useful data interchange within SAA components by providing services ranging from managing data links to specifying data streams for user applications. This paper discusses the role of Common Communications Support and the means for SAA users to access this support, and provides an overview of the functions and roles of various component architectures of Common Communications Support, along with their interrelationships.

On March 17, 1987, IBM embarked on an historic course to address consistency across its selected set of major computing environments. Systems Application Architecture (SAA) is a collection of selected software interfaces, conventions, and protocols that collectively, over time, will provide the framework for development and execution of consistent applications. These enhancements will pertain to consistency for the end-user interfaces and application-program interfaces across the future offerings of three major IBM systems: System/370, AS/400™, and Personal System/2®. SAA has also identified a set of common communication interfaces for interconnecting the three SAA computing environments. The end-user interface, application-program interface, and SAA applications are treated in other documents[1,2] and companion papers in this issue.[3–7] This paper addresses the Common Communications Support in SAA.

Systems Application Architecture. The primary objective of SAA is to present a consistent and cooperative system image across the identified computing environments. A consistent system image, in turn, implies ease in "porting" applications across such systems. As a secondary objective, cooperative operation of SAA systems implies that both the SAA applications and the data can be distributed among the SAA systems and effectively accessed by end users. Furthermore, these objectives should be achieved, over time, irrespective of whether SAA systems are located in a single enterprise within a country or within multiple enterprises participating in a global network.

Figure 1 depicts the components of an SAA system from a user perspective. One or more operating systems exist for each of the three SAA computing environments. These operating systems would, in turn, support a set of identified programming languages and language interfaces. The Common Communications Support (CCS) specifies protocols for interconnection and data interchange among SAA systems. The Common Programming Interface includes elements that provide access to the underlying CCS protocols. An end user can also access these protocols through a well-defined end-user interface

Figure 1 Structure of Systems Application Architecture

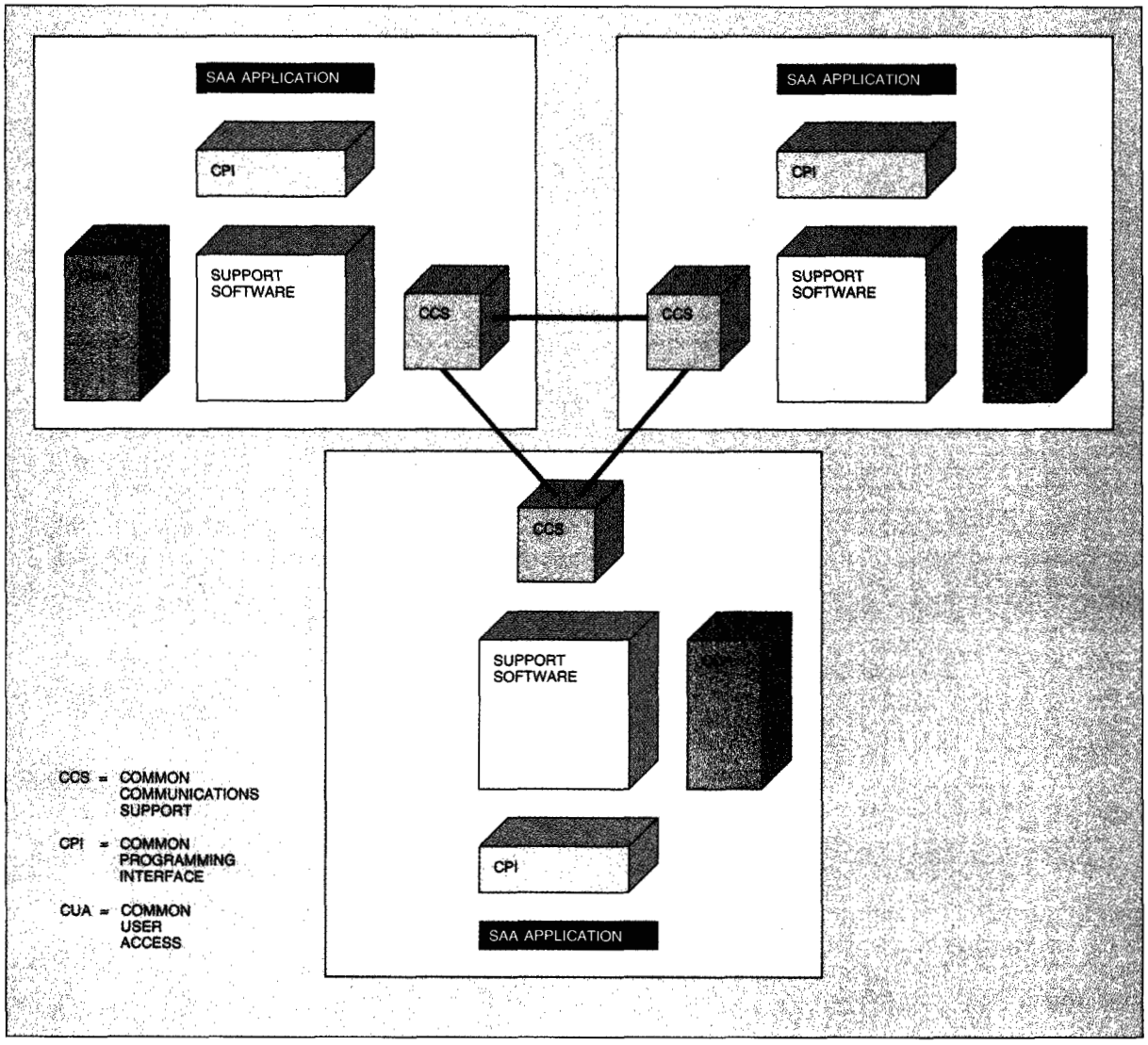


called the SAA Common User Access. This interface provides consistency for end-user interactions with a display terminal.

The role of CCS in SAA is depicted in Figure 2. The figure shows that neither the end user nor an application program directly accesses the CCS protocols. Access to CCS protocols is provided through the Common Programming Interface (CPI) or the Common User Access (CUA).

The primary responsibility of CCS is to provide interconnection of SAA systems, as shown in Figure 2. Such interconnection is not simply the physical transmission link connection between SAA systems. CCS also specifies protocols to permit useful data interchange between SAA applications and end users. Thus, in effect, CCS specifies a set of communication protocols that collectively provide such services. Some other related objectives for CCS are to provide services to enhance user productivity, such as distrib-

Figure 2 SAA and its Common Communications Support



uted data, program-to-program communication, and network management, and to permit access from non-SAA systems to SAA environments. To support this access, CCS includes international standards such as CCITT (International Telegraph and Telephone Consultative Committee) Recommendation X.25, and IEEE (Institute of Electrical and Electronics Engineers) Standards 802.2 for Logical Link Control and 802.5 for Token Ring. In addition, CCS specifies the formats and protocols of its elements, so that a non-SAA system may generate a compatible stream of data and operate with SAA systems.

SAA evolution and Common Communications Support. The evolution of SAA began with its announcement in March 1987. At that time, IBM identified SAA as the framework in which to develop consistent SAA applications across the future offerings of System/370, AS/400, and Personal System/2.

Figure 3 depicts the SAA evolution. The first goal of SAA was to present consistent interfaces for SAA systems, including ease in porting SAA applications across IBM's major computing environments. At that time, CCS specified a set of communication protocols

as the basis for interconnecting SAA systems. The second goal of SAA was to address distribution of applications and data. Here parts of an SAA application are distributed and executed cooperatively among more than one SAA system. The third goal of SAA is to present the capability of transparent user access to distributed data. The CCS part of SAA specifies protocols that support transparent local/remote data access.

In short, Common Communications Support provides interconnect and data interchange to SAA end users and applications through SAA interfaces, while also specifying the means by which non-SAA systems may participate in SAA environments.

The next section introduces the means available to SAA users for accessing CCS protocols. The succeeding section provides details of CCS protocols, including the supported international standards. That section also addresses the approaches for interconnection of non-SAA systems with SAA environments. The section after that uses an example of a hypothetical SAA Enterprise Information System to illustrate the various SAA elements and their role in the context of CCS.

SAA interfaces and Common Communications Support

In this section, we present the means available to an SAA user for accessing CCS functions.[8] An underlying CCS goal is to avoid exposing an SAA application to the details of CCS protocols and formats. Instead, an SAA application has available a collection of high-level language interfaces that, in turn, provide access to CCS protocols.

Figure 4 presents an abstract layered structure of SAA interfaces as they relate to CCS. An end user can be an operator who may use an IBM Personal Computer or Personal System/2 workstation to interact with an SAA system. The means for interaction available on such workstations are the keyboard and the associated display screen. A set of rules has been published that defines the use of function keys, screen management, and the like under SAA.[2] These rules

Figure 3 SAA evolution and role of Common Communications Support

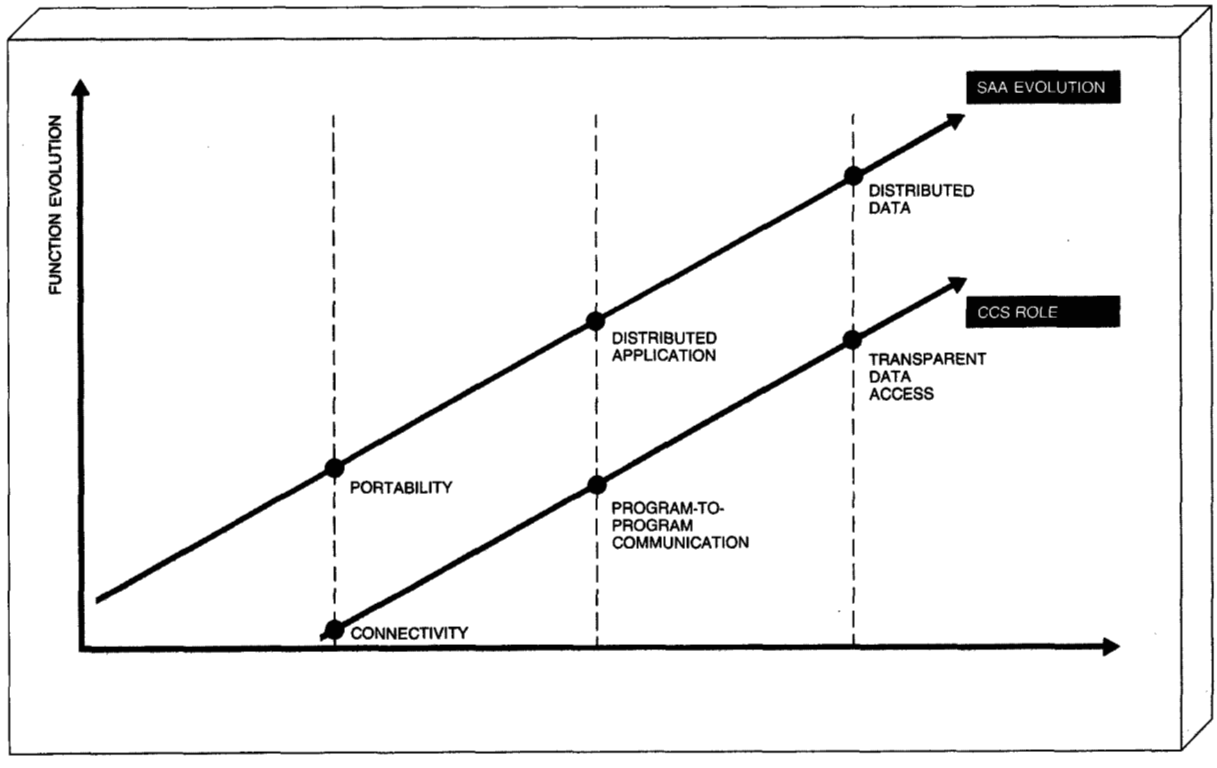
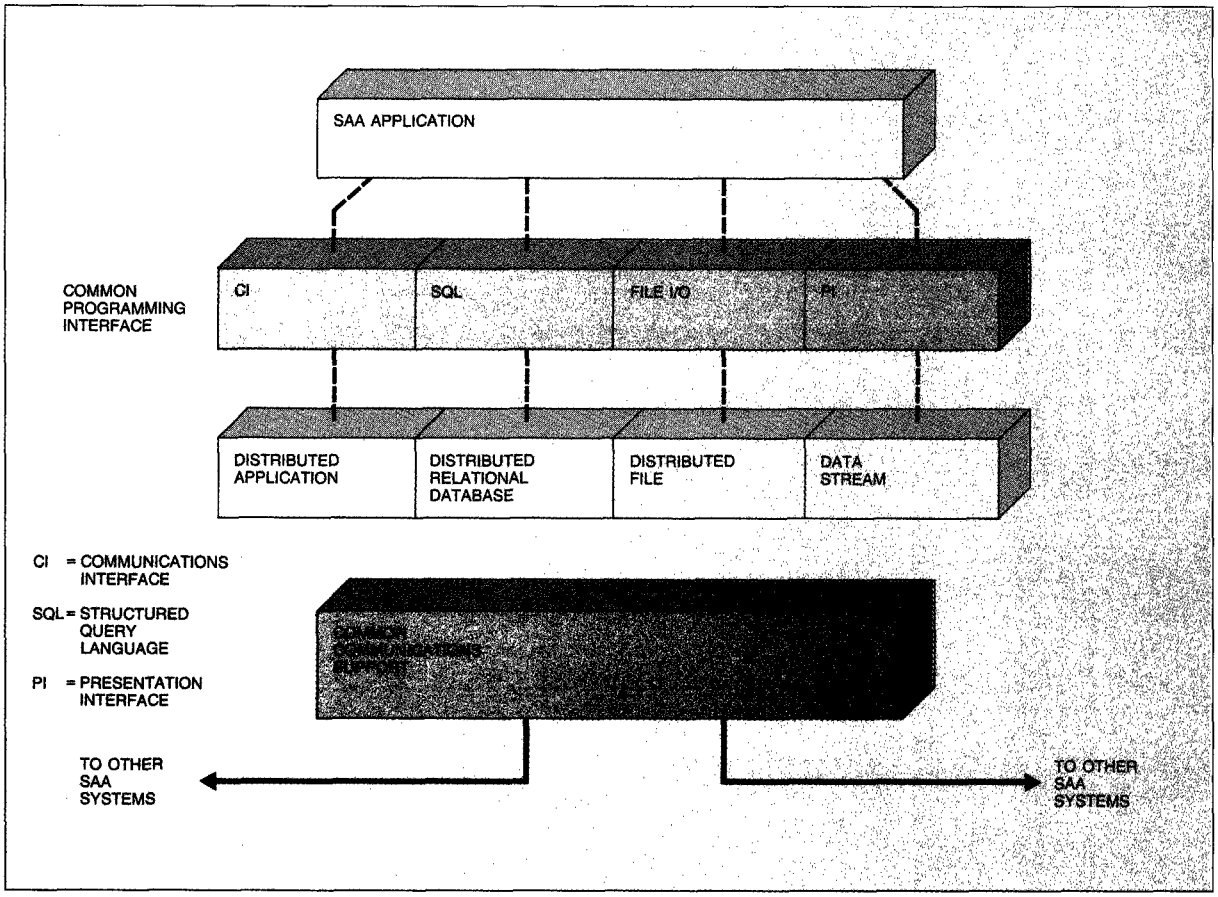


Figure 4 SAA application access to CCS functions



permit a consistent end-user interface across the SAA applications.

An SAA application is a program written in an SAA high-level language, e.g., FORTRAN, C, COBOL, CSP, or REXX, that may reside in one or more SAA systems. A given SAA application may require access to certain CCS protocols through the use of the Common Programming Interface. This is shown in Figure 4 and explained below.

Distributed applications: Program-to-program communication. If an SAA application needs to access another SAA application and exchange data, this communication is accomplished through the use of a collection of high-level language statements. These statements are defined as the Communications Interface, which is part of the CPI of SAA.[9] At program execution time, these statements result in generating appropriate CCS formats and protocols to accomplish communication with another SAA application. CCS supports this interface by utilizing the SNA Logical Unit Type 6.2 protocol.[10] A simple exchange using the Communications Interface of CPI could be to start a conversation between two programs, send data, and then terminate the conversation.

Distributed data: Files and databases. SAA applications may also require access to data that may reside either in the local SAA system or in a remote SAA system. Data may be stored, retrieved, and updated in one of two ways: as sequential, direct, or keyed-access files, also known simply as "files," or as a relational database. SAA has specified elements of the CPI to support access of remote data.

For data stored as files, the CPI element comprises the existing high-level language constructs for file input and output. Distributed data access for files is accomplished as follows. During program execution time, each data access is received by the underlying software supporting the file system. The file access software determines the location of the file by using side directory information that relates files to their locations. If the data reside in a local system, the access request is handled simply by the local file management system. If the data reside in another system, the data access request is transformed to appropriate CCS protocols and sent to the remote system. The CCS protocols used to accomplish these functions are IBM's Distributed Data Management (DDM) architecture and SNA Logical Unit Type 6.2, as described later.

CCS specifies a set of protocols for interconnection and communication among SAA systems.

The CPI element for relational database is IBM's Structured Query Language (SQL). This language provides access to computer data with a well-organized (structured) set of requests and queries.[11,12]

SNA Logical Unit Type 6.2, as described in the next section, is used for the session protocols.

Presentation Interface: Displays and printers. The SAA Presentation Interface defines the interface for programmers and users with a comprehensive set of functions that allow information to be displayed or printed in an effective manner. The major functions provided are windowing, support to enable applications to conform to SAA Common User Access, graphics, fonts, and double-byte character sets.[13] An SAA application uses the Presentation Interface for presenting and receiving data from an application user. CCS has specified two data streams, one each for printers and for display terminals, as described in the next section.

We have outlined the process by which SAA applications or end users may access CCS protocols. In each case, the SAA application is presented with a high-level interface, without being exposed to the details of underlying CCS protocols. In the following section, we address the specific CCS protocols for SAA, including those that support the above elements of the CPI.

Common Communications Support

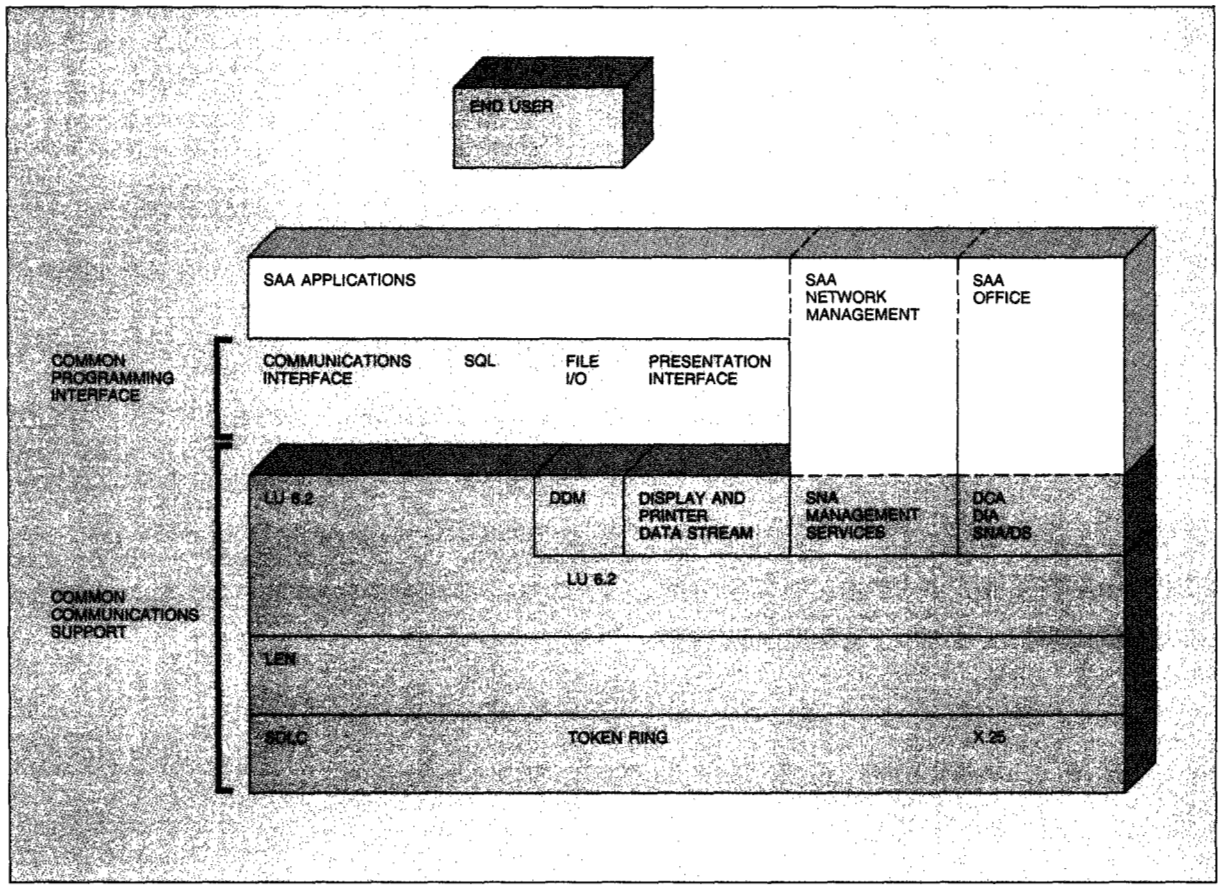
Common Communications Support specifies a set of protocols for interconnection and communication among SAA systems. SAA applications access these protocols through the CPI. As noted in the previous section, the CPI presents a high-level programming interface to the SAA applications, thereby shielding the SAA application programmer from the details of CCS protocols. In this section, we present an overview of CCS protocols.

The makings of Common Communications Support. An SAA system may connect to another SAA system, as well as to devices such as displays and printers. An SAA system may also require access from a non-SAA system. Thus, CCS includes architectures ranging from Data Link Control to application services and data streams, as well as selected international standards.

CCS is constituted primarily of elements of SNA. Selected international standards such as CCITT recommendations and IEEE standards have also been included.

Elements of CCS have been classified in five broad categories, namely data streams, application services, session services, network, and data link controls. These broad categories correspond roughly to groupings of SNA and Open Systems Interconnection (OSI) layers, although no attempt is made to establish any layer-to-layer correspondence. Figure 5 depicts the CCS protocols and their potential relationship to the CPI. In certain cases, such as document distribution and network management, the SAA application may be potentially integrated with the underlying CCS architectures. Any requirement for a CPI element for such services will be addressed as part of potential SAA enhancements.

Figure 5 Common Communications Support



In the remaining part of this section, we first describe the protocols for CCS and then discuss the interworking of SAA and non-SAA systems. Detailed descriptions of each of the identified CCS protocols can be found in the referenced publications.

Common Communications Support for Presentation Interface: Data streams. The Presentation Interface element of CPI permits an SAA application to write, in a consistent fashion, statements that result in generating a data stream for a printer or a display terminal. A data stream is a continuous ordered stream of data elements conforming to a given format. CCS has identified the data streams for displays and printers, namely the 3270 Data Stream and the Intelligent Printer Data Stream.

The 3270 Data Stream, also referred to as the IBM 3270 Information Display Data Stream, is a formatted data stream used for transmitting data between an application program and a terminal.[14] The data stream controls the processing and formatting of data by using commands, attention identifiers, and structured fields. An outbound data stream is sent from an SAA application to a device, and an inbound data stream is sent from the display terminal to the SAA application. The commands control such things as whether the SAA application writes to or reads from the display terminal, and whether the screen is erased before new data are written. The write control character, as shown in Figure 6, is used to provide such functions as sounding the alarm and enabling the keyboard. An attention identifier describes the action that caused the inbound data stream to be transmitted. Structured fields are used to transmit images and graphics and to convey additional control functions and data to or from the terminal.

The Intelligent Printer Data Stream (IPDS) is the host-to-printer data stream for all-points-addressable (APA) printing.[15] It provides commands to transmit APA data, along with a set of interactive controls used to communicate between a host printer driver code and the microcode in the printer. The commands and controls are defined in self-identifying structured fields which are separate from any carrying protocols. The APA capability of IPDS makes possible the presentation of pages containing a mixture of different data types: high-quality text, image, vector graphics, and bar code. Interactive support in IPDS allows for the dynamic management of resources downloaded to a printer, control of device functions such as duplexing, and media-bin selection. IPDS also provides for synchronization of the host print process and the printer in order to recover from errors.

Common Communications Support for distributed office. In an office environment, an SAA application is integrated with the underlying CCS architectures, as shown in Figure 5. Here the CCS support is provided through three architectures: Document Content Architecture, Document Interchange Architecture, and the SNA Distribution Services architecture. These architectures are briefly described below.

The Document Content Architecture (DCA) defines the structure and meaning of the content of a document that can be interchanged among office systems. SAA specifies DCA for revisable-form-text data stream (RFT-DCA) for document definition. A revisable-form-text data stream consists of format units followed by text units and an end unit, as shown in Figure 7. The format units provide information that may pertain to the entire document, and contain parameters that specify information such as interpretation of graphic characters in the document, identifiers for the dictionary that can be used to assist in spelling verification, punctuation formats, and composition and formatting for lines and pages. There may be one or more text units, with each text unit representing a page. The end unit specifies the end of a revisable-form-text data stream. RFT-DCA is described in detail in Reference 16.

In an SAA office system, a given document is formatted using the RFT-DCA data stream and can be interchanged with other SAA office systems, using Document Interchange Architecture (DIA), which in turn uses SNA Distribution Services. DIA defines the functions for interchanging documents between separate office systems.[17] DIA provides a variety of services for

Figure 6 Common Communications Support for display terminals and printers

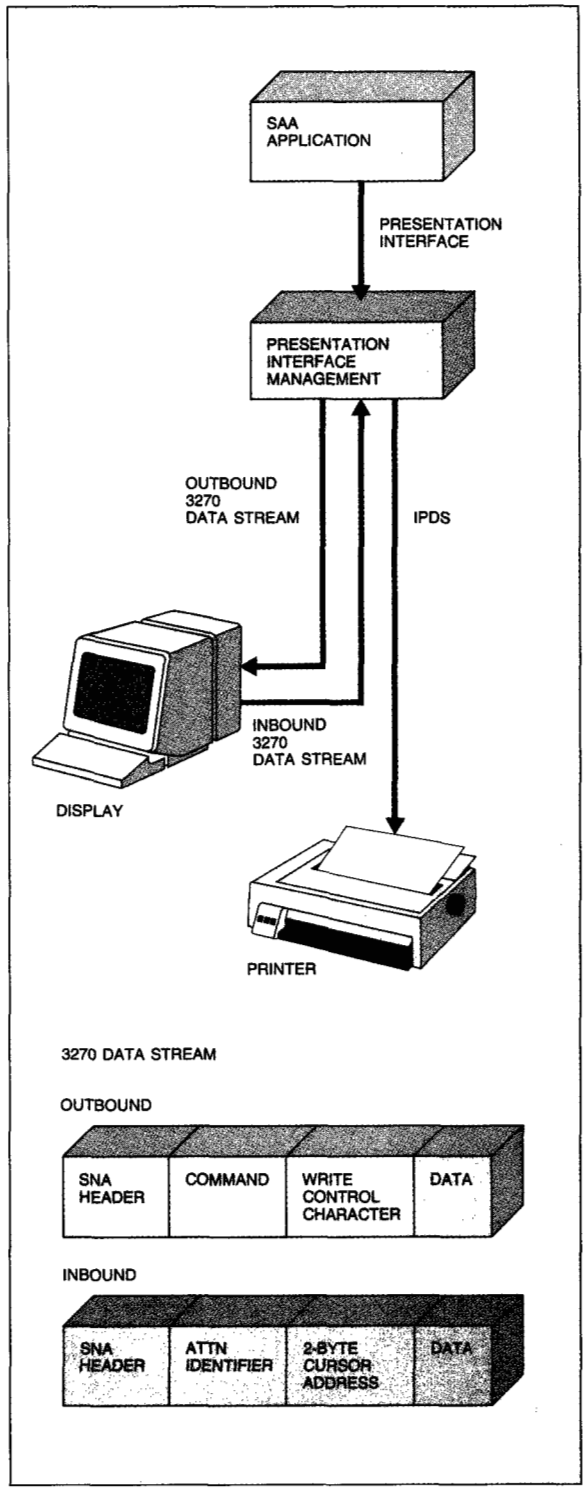
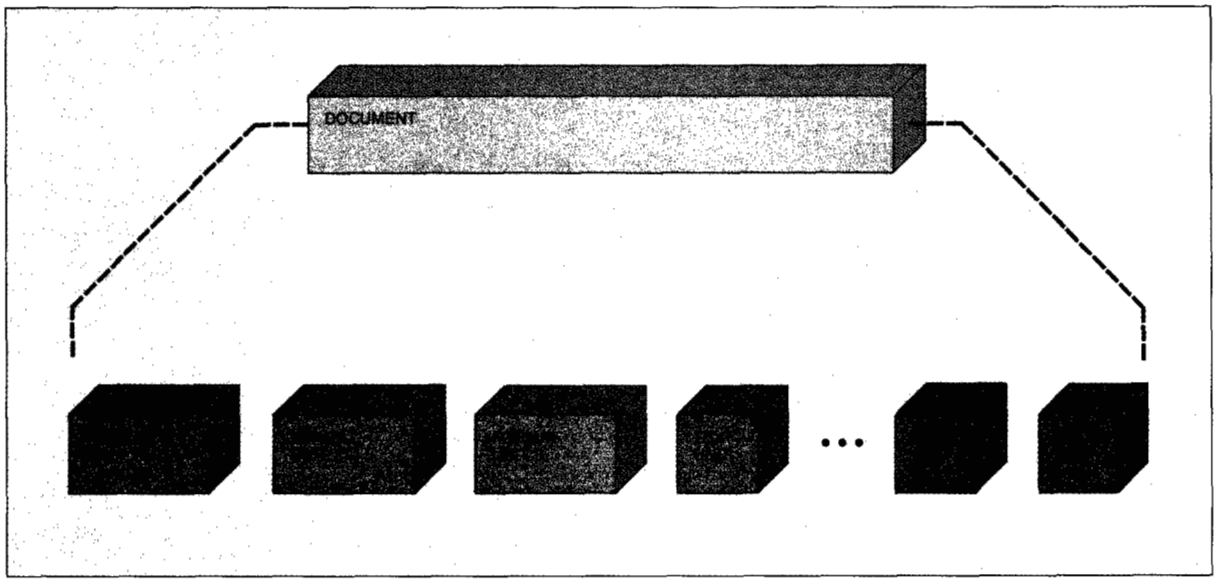


Figure 7 Revisable-form-text data stream



accomplishing the interchange. For example, the document library services of DIA allow documents to be filed, searched, retrieved, or deleted from the library. The document distribution service of DIA provides the means to deliver a document to other office systems. This distribution is accomplished by utilizing SNA Distribution Services (SNA/DS), as explained below.

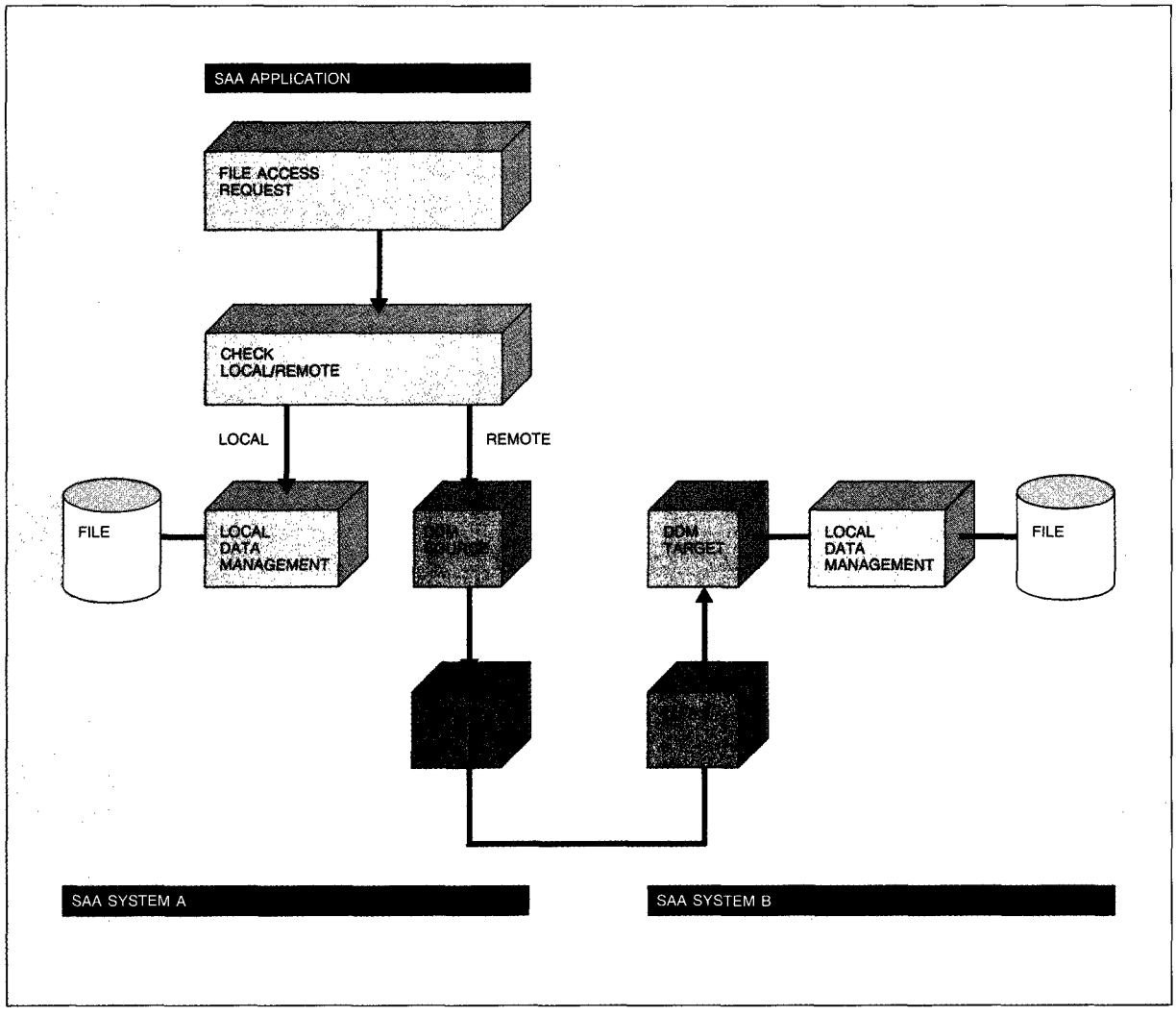
SNA/DS, the store-and-forward facility of SNA, uses SNA Logical Unit Type 6.2 sessions to distribute data. However, it does not require the prior establishment of all end-to-end sessions. If a given destination node does not have a session (or is not even powered on), SNA/DS delivers the data to a point closest to the destination node. This function is also referred to as asynchronous delayed-delivery or connectionless data distribution. Thus the sender, the receiver, and the requisite communication resources need not be active at the same time. SNA/DS allows inclusion of multiple destinations in its header. Since the data are staged as they progress through the network, SNA/DS uses its distribution capability on the basis of the network topology to deliver data to multiple destinations. SNA/DS can be used for documents as well as for messages and files.[18,19]

Common Communications Support for network management. Network management is the process of planning, organizing, and controlling a network. The CCS network management architecture is integrated with the SAA network management system. As such, SAA network management presents an end-user interface to its users but does not use any element of CPI. There is a requirement to provide a common end-user interface for network management across SAA systems.

SAA network management architecture is part of SNA Management Services, which provide services in the following areas: Problem Management, Configuration Management, and Performance and Accounting Management. In short, Problem Management pertains to dealing with a problem from its detection through its resolution. The steps of Problem Management are (1) determination, (2) diagnosis, (3) bypass and recovery, (4) resolution, and (5) tracking and control. Configuration Management consists of the facilities and the process necessary to plan, develop, operate, and maintain an inventory of information system resources, attributes, and relationships. Performance and Accounting Management is the process of quantifying, measuring, reporting, and controlling the usage, responsiveness, availability, and cost of a network.[7,20]

Common Communications Support for distributed data. In the last section, we outlined the process and interfaces used by SAA applications to access data located within the local SAA system or in a remote SAA system. CCS provides the services to accomplish

Figure 8 Distributed files



creation, access, update, and deletion of remote data from a given SAA system. These services are provided transparently to the SAA application. That is, while SAA application programs access data as though the data were local, distributed data services are invoked to access data that reside in a remote location. We address two kinds of distributed data here. First, such record files as sequential, direct, or indexed-sequential are classified as "flat file data," or simply "files." The other kind of record files, relational or hierarchical database, are simply referred to as "database." In either case, the underlying connection support is provided through SNA Logical Unit Type 6.2.

Distributed file processing is accomplished by using the Distributed Data Management (DDM) architecture. For example, consider a file access by an SAA application in SAA System A, as shown in Figure 8. First, it is determined whether the file is located locally in SAA System A or in some remote system. Access to a local file is passed to and processed in the local data management system. If the file is located in a remote system, the request is forwarded to the DDM Source process. The DDM Source creates appropriate commands and forwards these commands to the DDM Target process in the appropriate SAA system. Let us assume that the file resides in SAA System B. Then the DDM Target routine in SAA

System B handles this request in conjunction with its local data management system. Appropriate responses or file data are sent back from SAA System B to SAA System A using the same path in the reverse direction.

DDM architecture can be viewed as consisting of three layers: the Data Manager, the Agent, and the Communication Manager. The Data Manager provides functions that pertain to both files and records in the files. Files on remote systems can be created, deleted, renamed, locked, or unlocked, and their attributes can be retrieved. Sets of records can be loaded into or unloaded from a remote file for a whole file transfer. Individual records of remote files can be read, written, modified, or deleted, either randomly or consecutively, by record number or by key value. For every application program, a DDM Agent is provided. A DDM Agent performs all of the necessary conversions and parsing of DDM data. The DDM Agent also enforces security and provides any cleanup or recovery processing. The Communication Manager interfaces with the underlying Logical Unit Type 6.2 protocols to accomplish data interchange between SAA systems.

For distributed SAA applications, each component of an application uses the Communications Interface.

CCS for distributed database provides protocols that permit an SAA application to include the same Structured Query Language (SQL) statements to access or process a database, whether the database is local or remote. This is accomplished by utilizing SNA Logical Unit Type 6.2.

Common Communications Support for distributed applications: LU 6.2. CCS for distributed SAA applications is provided through SNA Logical Unit Type 6.2 (LU 6.2), which defines the formats and protocols for general-purpose program-to-program communication.

For distributed SAA applications, each component of an application uses the Communications Interface (CI), which is an element of the CPI.[9] SAA applications include CI statements that access subroutines in the system. These subroutines, in turn, access the code that provides LU 6.2 formats and protocols. Since LU 6.2 is the session-level support for SAA applications, LU 6.2 is also used to support other distributed SAA functions such as distributed data and network management.

Programs using LU 6.2 communicate via "conversations." The LU 6.2 conversation functions are defined in terms of programming-language-like statements called "verbs." The CI statements may be mapped directly to these verbs. LU 6.2 includes a set of base verbs as well as some optional capabilities for enhanced services. These verbs, along with relevant parameters, provide functions such as starting and ending conversations between remote programs, sending and receiving data messages, synchronizing processing between programs, and notifying a partner program of any errors.[10,21]

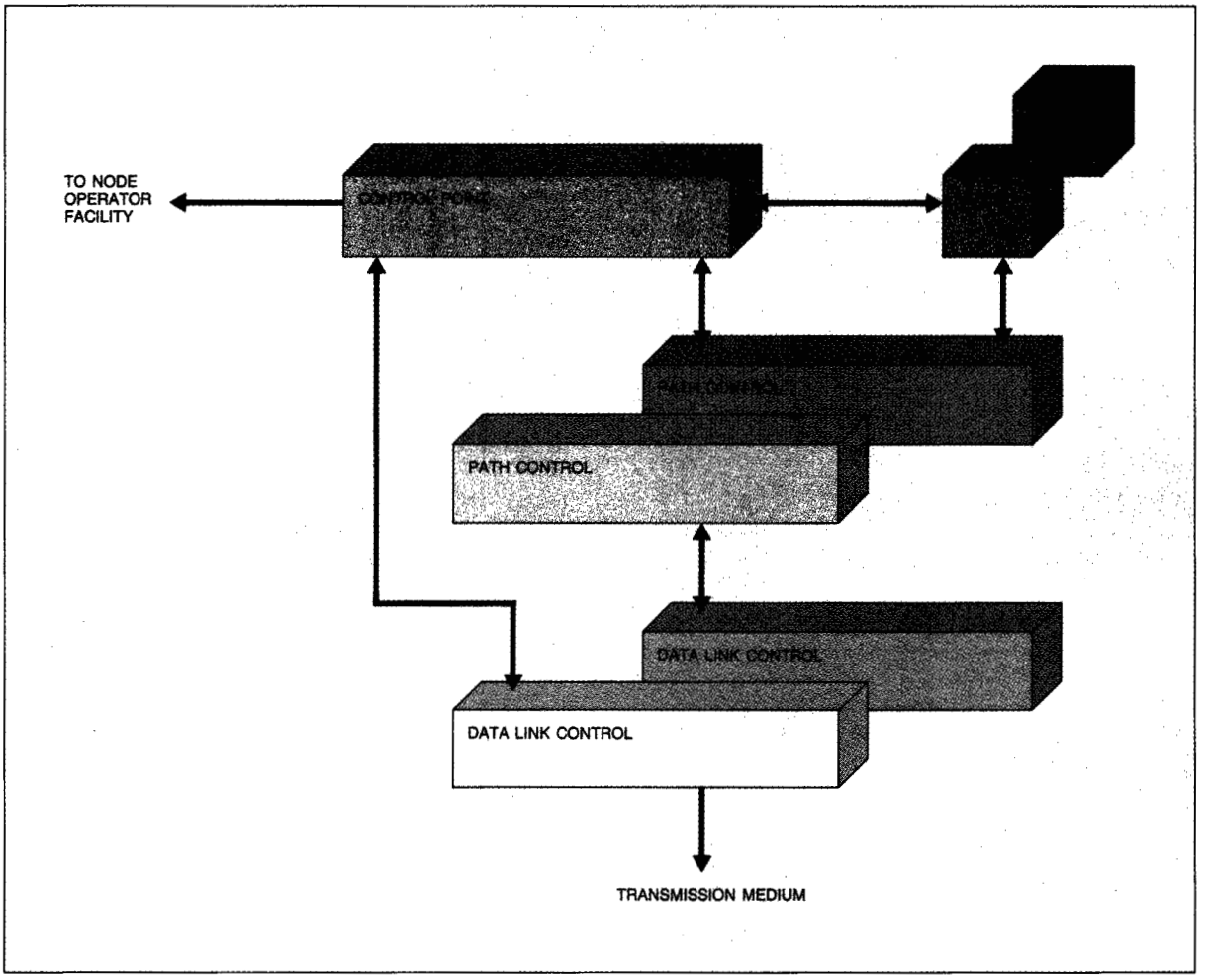
The LU 6.2 protocols use underlying SNA Path Control and Data Link Control to receive and transmit data among SAA systems.

Common Communications Support: Network. For network-level support, CCS specifies SNA Low Entry Network (LEN) or SNA Type 2.1 protocols. Type 2.1 protocols allow peer-to-peer connection of distributed processes, as well as providing the physical and logical connectivity required to support LU 6.2 sessions.

As shown in Figure 9, a Type 2.1 node consists of a control point, multiple instances of logical units (LU 6.2), path control, and data link control (DLC). The control point manages the resources of the Type 2.1 node. It creates the path control and DLC instances, directs activation and deactivation of a link, and assists LU 6.2 in session initiation and termination. Path Control delivers message units between LUs in the same or different nodes. Path Control also performs segmentation and reassembly of message units. The DLC used between two Type 2.1 nodes may be IBM Token Ring, CCITT Recommendation X.25, or Synchronous Data Link Control (SDLC), as described below. Formats and protocols for Type 2.1 nodes are presented in detail in Reference 22.

Common Communications Support: Data link control. SAA systems may be interconnected using local area networks, telecommunication links, or packet-

Figure 9 Structure of LEN node

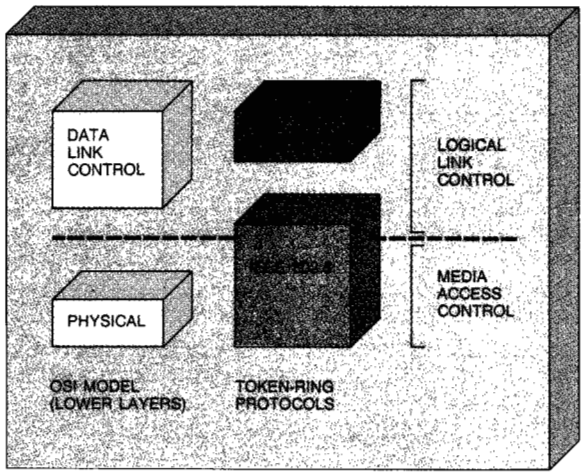


switched networks. For data link control, SAA specifies three protocols, namely CCITT Recommendation X.25, SDLC, and Token Ring. Any of these three protocols may be used to interconnect SAA systems, depending on the respective means of interconnection. Whereas SDLC and Token Ring primarily support SNA attachments, X.25 permits attachment of SAA as well as non-SAA systems to access and participate in SAA. The three data link controls are briefly described next.

CCITT Recommendation X.25 has evolved over the last decade, along with numerous public and private packet-switched data networks. Within an SAA network, an SNA node can communicate with another SNA node using X.25 virtual circuits. In this case, X.25 virtual circuits are mapped to SNA data links. Since more than one SNA session can map to an SNA link, more than one SNA session may be mapped to a given X.25 virtual circuit. Some SNA nodes, such as clusters and terminals, attach to only one adjacent SNA node using a single physical link. Such a single link can be treated as a single virtual circuit.

SNA nodes that can connect to multiple adjacent nodes over different physical links, or multidrop links, can be connected using multiple virtual circuits. SNA nodes interconnected by virtual circuit services remain logically adjacent, and the virtual circuit protocols provide the mechanism to exchange

Figure 10 Lower layers of OSI model and Token-Ring protocols



data between these nodes. As such, X.25 virtual circuits provide functions similar to those of other data transmission facilities.[23–26] CCS has specified a set of protocols and architectures, as shown in Figure 5, for exchanging data above the X.25 packet level. CCITT Recommendation X.25 may also be used to exchange data between an SAA system and a non-SAA system, as described later in this section.

SDLC defines formats and protocols for transferring data across a transmission link. It prescribes a discipline for managing synchronous, code-transparent, serial-by-bit information transfer between nodes that are joined by telecommunication links. SDLC is described in detail in Reference 27.

The IBM Token-Ring protocols support the data link control and the physical layer, as shown in Figure 10, and are described in detail in References 28 and 29. The IEEE 802.2 standard specifies Logical Link Control (LLC). LLC defines formats and protocols for exchanging frames between LLC layers attached to a local area network. It has provisions such that only error-free, nonduplicate, properly ordered frames are delivered to the data link user. IEEE standard 802.5 specifies a ring using a token-passing scheme for access. This part of the token-ring protocols pertains to the local area network hardware and encompasses the Physical Layer of the Reference Model for OSI. Since this standard specifies the access to transmission media of token-ring local area networks, the standard is called Media Access Controls for Token Ring.

Participation of non-SAA systems with SAA networks. A non-SAA system can participate in an SAA network in one of two ways. First, CCS has specified a list of IBM publications that describe details of each element of CCS. A given non-SAA system may generate the appropriate data stream based on these architectures and may exchange data with SAA systems.

A non-SAA system may also utilize CCITT Recommendation X.25 for access to SAA systems. Here we outline an approach for such access. In this approach, each SNA session is mapped one-on-one to an X.25 virtual circuit. As shown in Figure 11, an SAA application is serviced by LU 6.2 protocols. At the node attaching to the packet-switched data network, an LU 6.2 session is mapped to an X.25 virtual circuit. The attaching node also presents the appearance of X.25 Data Terminal Equipment (DTE). If the target DTE is an X.25 DTE, the X.25 virtual circuit would exist between the SNA node adjacent to the public packet-switched network and the X.25 DTE. For some non-X.25 DTEs, a Packet-Assembler Disassembler (PAD) may be used to convert the non-X.25 protocols to X.25 protocols. (CCITT Recommendations X.3, X.28, and X.29 specify the PAD functions.) In either case, the higher-level protocols that interchange data with the SAA application need to be those specified in the above description of CCS.

An SAA Enterprise Information System and its Common Communications Support

The preceding section described the role of individual CCS architectures in achieving the objective of interconnection and data communication among SAA systems. In this section, we use a hypothetical example of an SAA Enterprise Information System to illustrate the role of some of these architectures as well as the role of other SAA components.

An SAA Enterprise Information System consists of one or more SAA systems interconnected using CCS; it supports CUA for end-user interfaces and CPI for SAA applications. In the SAA Enterprise Information System shown in Figure 12, an end user on a Personal System/2 (PS/2®) workstation is accessing an SAA application. The interactions between the user and the PS/2 display conform to CUA. In this example, the SAA application, residing in the PS/2, requests access from a file. If the requested file is located in the (local) PS/2, the request is processed by the local file management system. If the file is located in a remote system, such as in the System/370 with the

Figure 11 Attachment of non-SAA DTEs to SAA system using CCITT Recommendation X.25

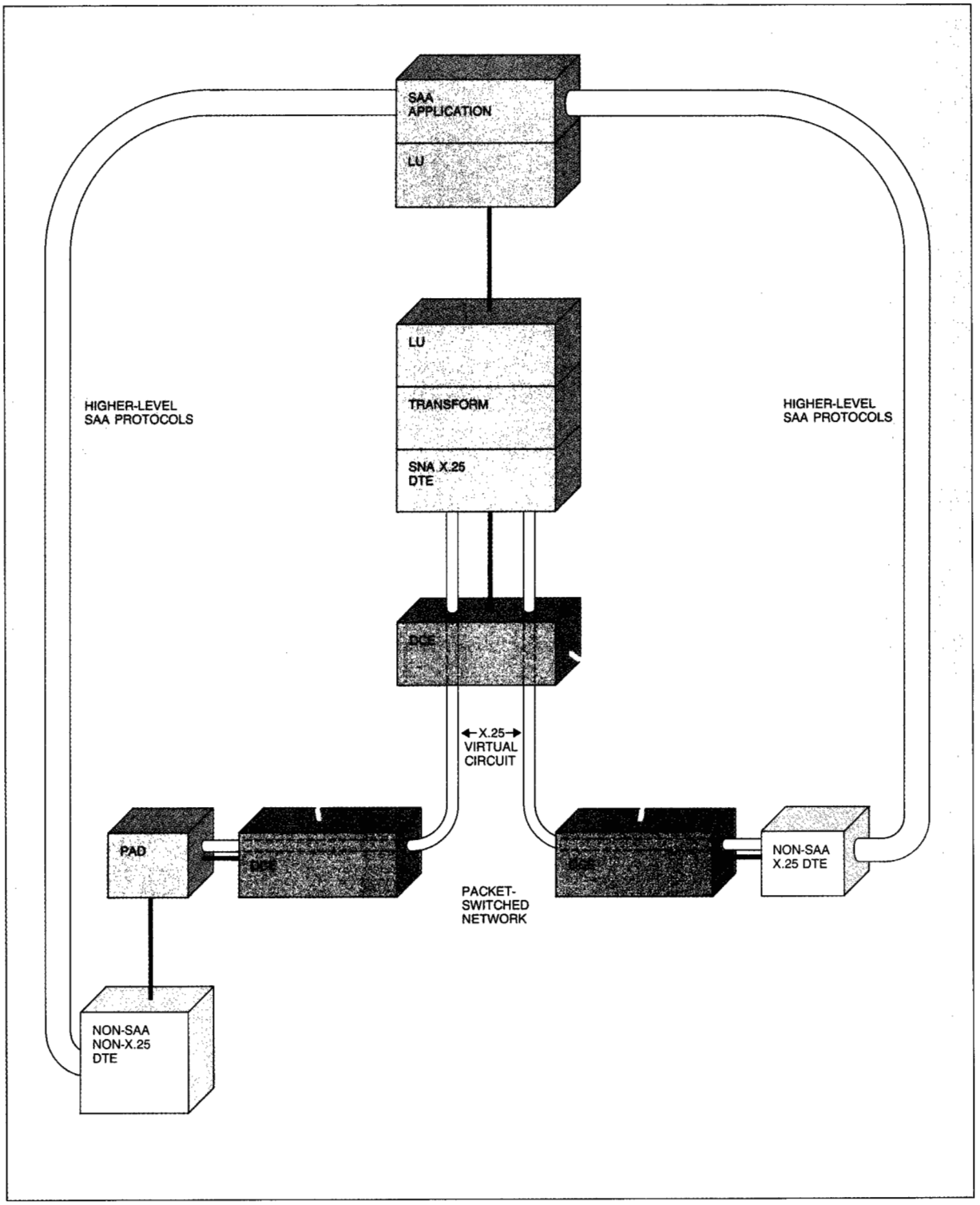
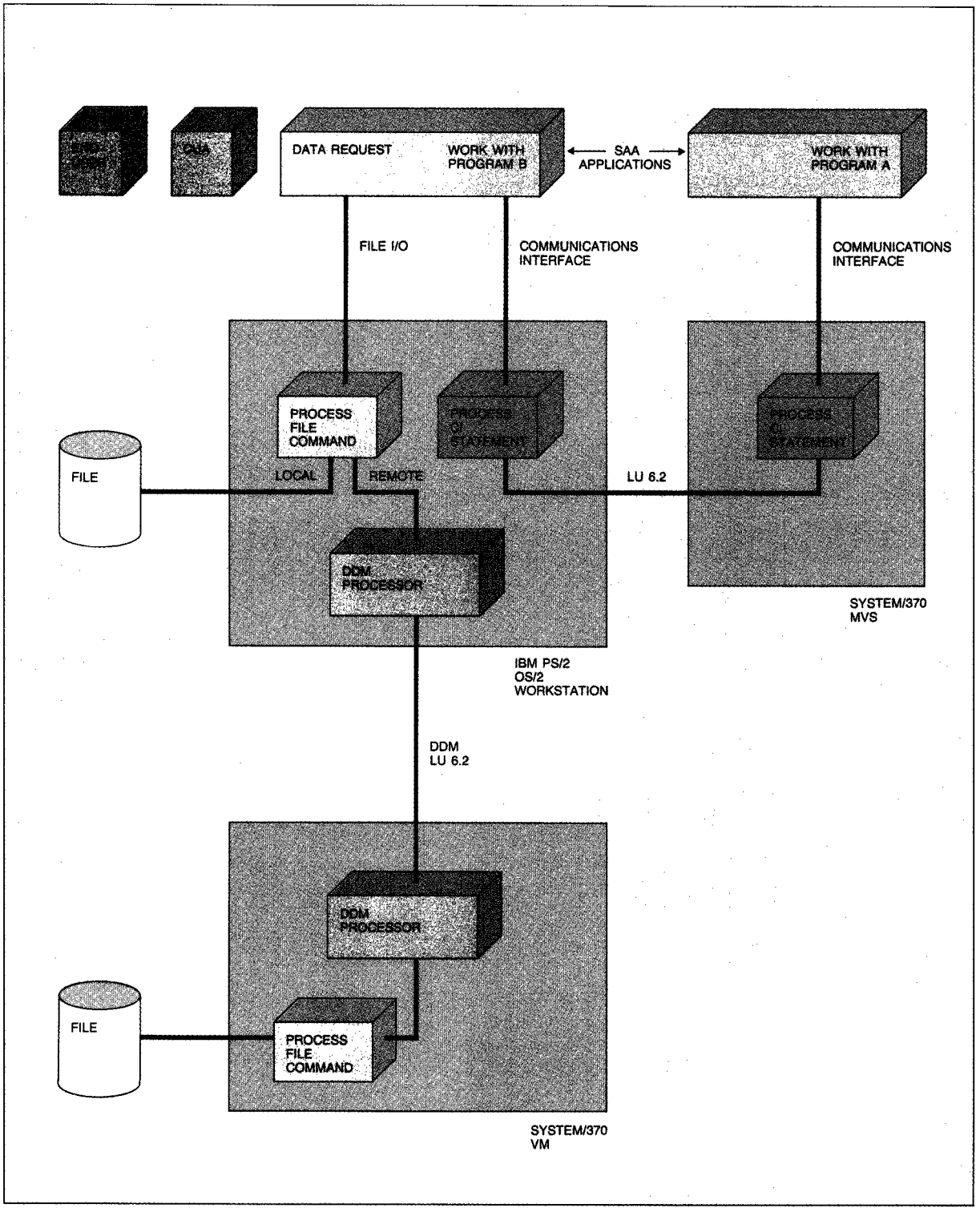


Figure 12 An SAA Enterprise Information System



Virtual Machine (VM) operating system, the request is handled by the local DDM Processor. The DDM protocols and the underlying LU 6.2 protocols are used to format and send a request to the remote System/370 for execution. The results of the request are received back in the PS/2 and presented to the SAA application. Note that the SAA application is not exposed to the location of the file or to the details of DDM or LU 6.2 protocols in this case.

The SAA application in the PS/2 workstation may also access another SAA application in another SAA system, for example, in the System/370 with the Multiple Virtual Storage (MVS) operating system. Here, the SAA application uses the Communications Interface element of the CPI to communicate with the remote SAA application. The underlying software support for the CPI converts these requests to LU 6.2 protocols, which in turn initiate, manage, and terminate this program-to-program communication. The two SAA applications may be independent applications or two parts of a single distributed SAA application.

Similar scenarios exist for SAA applications using distributed relational data and the other SAA systems (for example, AS/400), thus providing users with a great deal of flexibility in building their SAA Enterprise Information System.

Summary

Systems Application Architecture is IBM's framework for consistency across its three major computer systems. Common Communications Support addresses the interconnection and data communication functions among the SAA systems and the attachment of non-SAA systems. CCS consists of selected SNA architectures and international standards. As SAA expands over time, the role of CCS will correspondingly be extended. Additional requirements for CCS extensions include file services and support for Open Systems Interconnection.

Acknowledgments

Several individuals have helped me over the course of writing about this diverse topic. I owe special gratitude to John Marsland, George Deaton, Don Holtz, John Drake, Dave Rose, Rich Demers, Robert Pascoe, George Zagelow, Carol Berinato, Jane Munn, Jim Frey, Roger Reinsch, Jeff Stark, Hamid Khafagy, Alan Ganek, Homer Leonard, Karla Norsworthy, and Judy Haber. For editorial assistance, special thanks are due to the staff of the *IBM Systems Journal.*

Cited references and note

1. *Systems Application Architecture—An Overview*, GC26-4341-0, IBM Corporation (May 1987); available through IBM branch offices.
2. *Systems Application Architecture Common User Access Panel Design and User Interaction*, SC26-4351-0, IBM Corporation (December 1987); available through IBM branch offices.
3. R. E. Berry, "Common User Access—A consistent and usable human–computer interface," *IBM Systems Journal* **27**, No. 3, 281–300 (1988, this issue).
4. A. L. Scherr, "SAA distributed processing," *IBM Systems Journal* **27**, No. 3, 370–383 (1988, this issue).
5. L. A. Buchwald, R. W. Davison, and W. P. Stevens, "Integrating applications with SAA," *IBM Systems Journal* **27**, No. 3, 315–324 (1988, this issue).
6. D. E. Wolford, "Application enabling in SAA," *IBM Systems Journal* **27**, No. 3, 301–305 (1988, this issue).
7. D. B. Rose and J. E. Munn, "SNA network management directions," *IBM Systems Journal* **27**, No. 1, 3–14 (1988).
8. Since the focus of this paper is on Common Communications Support, only those elements of the Common Programming Interface are described that relate to CCS. A detailed treatment of the Common Programming Interface is given in Reference 6.
9. *Systems Application Architecture Common Programming Interface Communications Reference*, SC26-4399, IBM Corporation (May 1988); available through IBM branch offices.
10. *Systems Network Architecture Transaction Format and Protocol Reference Manual Architecture Logic for LU Type 6.2*, SC30-2369, IBM Corporation; available through IBM branch offices.
11. *Systems Application Architecture Common Programming Interface Database Reference*, SC26-4348, IBM Corporation (September 1987); available through IBM branch offices.
12. R. Reinsch, "Distributed database for SAA," *IBM Systems Journal* **27**, No. 3, 362–369 (1988, this issue).
13. *Systems Application Architecture Common Programming Interface Presentation Reference*, SC26-4359-0, IBM Corporation (October 1987); available through IBM branch offices.
14. *IBM 3270 Information Display System Data Stream*, GA23-0059-3, IBM Corporation (1986); available through IBM branch offices.
15. *Intelligent Printer Data Stream*, S544-3417, IBM Corporation (August 1987); available through IBM branch offices.
16. *Document Content Architecture: Revisable-Form-Text Reference*, SC23-0758-1, IBM Corporation (1986); available through IBM branch offices.
17. *Document Interchange Architecture: Technical Reference*, SC23-0781, IBM Corporation; available through IBM branch offices.
18. B. C. Housel and C. J. Scopinich, "SNA Distribution Services," *IBM Systems Journal* **22**, No. 4, 319–343 (1983).
19. *Systems Network Architecture Format and Protocol Reference Manual: Distribution Services*, SC30-3098-2, IBM Corporation (July 1985); available through IBM branch offices.
20. *Systems Network Architecture Format and Protocol Reference*

Manual: Management Services, SC23-0757-1, IBM Corporation (March 1986); available through IBM branch offices.
21. J. P. Gray, P. J. Hansen, P. Homan, M. A. Lerner, and M. Pozefsky, "Advanced program-to-program communication in SNA," *IBM Systems Journal* **22**, No. 4, 298–318 (1983).
22. *Systems Network Architecture Format and Protocol Reference Manual Architecture Logic for Type 2.1 Nodes*, SC30-3422-0, IBM Corporation (December 1986); available through IBM branch offices.
23. *The X.25 Interface for Attaching SNA Nodes to Packet-Switched Data Networks: General Information Manual*, GA27-3345-2, IBM Corporation (March 1985); available through IBM branch offices.
24. *The X.25 1984 Interface for Attaching SNA Nodes to Packet-Switched Data Networks—Architecture Reference*, SC30-3409, IBM Corporation; available through IBM branch offices.
25. G. A. Deaton, Jr. and R. O. Hippert, Jr., "X.25 and related recommendations in IBM products," *IBM Systems Journal* **22**, Nos. 1/2, 11–29 (1983).
26. *The X.25 1984 Interface for Attaching SNA Nodes to Packet-Switched Data Networks—General Information Manual*, GA27-3761, IBM Corporation; available through IBM branch offices.
27. *IBM Synchronous Data Link Control Concepts*, GA27-3093, IBM Corporation; available through IBM branch offices.
28. N. C. Strole, "A local communications network based on interconnected token-access rings: A tutorial," *IBM Journal of Research and Development* **27**, No. 5, 481–496 (September 1983).
29. *IBM Token Ring Network Technology*, GA27-3732-0, IBM Corporation (1986); available through IBM branch offices.

Vijay Ahuja *IBM Communication Products Division, P.O. Box 12195, Research Triangle Park, North Carolina 27709.* Dr. Ahuja is currently the manager of Architecture and Performance in Advanced Teleprocessing Products. He received an M.S. in computer science from the University of North Carolina at Chapel Hill. In 1976, he received a Ph.D. in computer science, also from the University of North Carolina. From September 1986 to April 1988, he worked on the SAA Common Communications Support. His current interests are in the areas of network routing, congestion control, and deadlocks. Dr. Ahuja is the author of a textbook, *Design and Analysis of Computer Communication Networks*, published by McGraw-Hill, and several papers on network design problems.

Reprint Order No. G321-5324.