IBM

Systems Application Architecture

GC31-6810-0

Common Communications Support Summary

IBM

Systems Application Architecture

GC31-6810-0

Common Communications Support Summary

Special Notices

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates.

Any reference to an IBM licensed program or other IBM product in this publication is not intended to state or imply that only IBM's program or other product may be used.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Commercial Relations, IBM Corporation, Purchase, New York 10577.

This publication may include references to microcode. Some IBM products contain microcode classified as Licensed Internal Code. Licensed Internal Code is provided under terms and conditions set forth in IBM agreements, such as the Agreement for Purchase of IBM Machines and the Agreement for Lease or Rental of IBM Machines.

IBM® and PROFS® are registered trademarks of the International Business Machines Corporation. The following trademarks of the International Business Machines Corporation are used in this book:

- Application System/400™
- AS/400™
- DisplayWrite™
- Enterprise Systems Architecture/370™
- ESA/370™
- GDDM™
- ImagePlus™
- NetView™
- Operating System/400™
- Operating System/2™
- OS/2™
- OS/400™
- Systems Application Architecture™

Contents

Figures

About This Book

This book summarizes the elements of Common Communications Support, a part of Systems Application Architecture. Common Communications Support (CCS) specifies protocols for interconnecting SAA systems and interchanging data among those systems. This book does not describe product implementations of CCS.

Who Should Use This Book

This book is intended for those who need an overview of Common Communications Support and others who develop or adapt a product to attach to an IBM network.

You should have a broad knowledge of data communication networks and how they work. In addition, you are expected to understand the basic concepts of Systems Application Architecture, Systems Network Architecture (SNA), and Open Systems Interconnection (OSI). If you need additional information on these subjects, the following publications are recommended:

- *Systems Application Architecture: An Overview, GC26-4341*

- *Systems Network Architecture: Concepts and Products, GC30-3072*

- *Systems Network Architecture: Technical Overview, GC30-3073*

- *International Standard Information Processing Systems—Open Systems Interconnection—Basic Reference Model, ISO 7498.*

The "Bibliography" on page 65 has a complete list of reference manuals.

How to Use this Book

This book is divided into three parts, with eight chapters.

- Part 1. Introduction

  - "Chapter 1: An Introduction to Common Communications Support" explains the importance of Common Communications Support and how it relates to other parts of SAA. It also gives an overview of the CCS categories and how they relate to each other. Those who have no interest in implementing CCS might find that reading Chapter 1 will suffice.

- Part 2. IBM Architectures in CCS

  This part contains six chapters which describe the IBM architectures in Common Communications Support. The description of each architecture contains a topic, **"Minimum Functions for ..."**. These topics discuss the minimum functions that must be implemented with the specific architecture to ensure SAA conformance; that is, to ensure that data can be exchanged with another SAA system implementing the architecture.

  - "Chapter 2. Objects" describes the object content architectures—the role that they play in CCS and how they relate to the data streams described in Chapter 3.

  - "Chapter 3. Data Streams" describes SAA data streams—their role in CCS and how they relate to object architectures and the application services described in Chapter 4.

- "Chapter 4. Application Services" describes the services that support distributed data files, network management, and document distribution.

- "Chapter 5. Session Services" describes SNA logical unit 6.2, which provides session services for architectures in the application services category or for application transaction programs.

- "Chapter 6. Network" describes the SNA Type 2.1 Node architecture, which provides direct connectivity between SAA systems.

- "Chapter 7. Data Link Control" describes Synchronous Data Link Control (SDLC) and X.25, both of which provide data link control for wide-area networks. It also describes the token-ring protocol for local area networks.

• Part 3. International Standards in CCS

This part contains a chapter that describes the Open Systems Interconnection (OSI) protocols in CCS.

- "Chapter 8. International Standards" describes the international standards that allow IBM networks to communicate with non-IBM networks.

Note: Throughout the manual, terms are *italicized* when they are defined.

Part 1. Introduction

Chapter 1. An Introduction to Common Communications Support

Systems Application Architecture (SAA) is a collection of selected software interfaces, conventions, and protocols that, over time, will provide the framework for development and execution of consistent applications. SAA will provide a consistent system image across the future offerings of the following environments:

- Enterprise Systems Architecture/370 (ESA/370)

- VM (System Product or Extended Architecture)

- Operating System/400 (OS/400)

- Operating System/2 (OS/2) Extended Edition (EE).

Environments other than these four can implement elements of SAA; however, IBM will include all of the appropriate elements of SAA in the environments listed above.

The Structure of Systems Application Architecture

SAA builds on IBM's approach for structuring product and software management. Figure 1 shows the software foundation that is used consistently in all SAA environments.
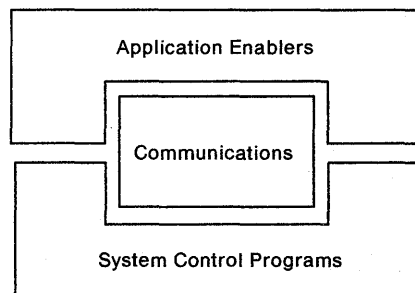
Figure 1. SAA Software Foundation

The following types of products are included in the software foundation:

- *Application enablers* represent code, such as compilers and database programs, needed by application programs.

- *Communications* represents code that allows the specific system to communicate with its attached devices, or with other systems in the network—VTAM in ESA/370 and VM, and the communications portions of OS/2 EE and OS/400.

- *System control programs* represent the operating system and its extensions for a given hardware environment.

Products in the software foundation differ from machine to machine; they manage and exploit the specific hardware in which they run.

With the SAA framework encompassing the software foundations of each hardware environment, applications can be developed that are portable across the hardware environments.

As seen in Figure 2, the SAA framework consists of interrelated components that frame the software foundation of IBM's application enablers, communications, and system control programs. These components—the Common User Access, the Common Programming Interface, and Common Communications Support— are the basis for developing common applications.

```
┌─────────────────────────────────────────────────────────┐
│                                                         │
│                   Common Applications                    │
│                                                         │
└─────────────────────────────────────────────────────────┘
┌──────────┐  ┌────────────────────────────────┐  ┌──────────┐
│          │  │  Common Programming Interface   │  │          │
│          │  └────────────────────────────────┘  │          │
│          │  ┌────────────────────────────────┐  │          │
│          │  │                                │  │          │
│ Common   │  │    ┌──────────────────────┐    │  │ Common   │
│ User     │  │    │  ┌────────────────┐  │    │  │ Communi- │
│ Access   │  │    │  │                │  │    │  │ cations  │
│          │  │    │  └────────────────┘  │    │  │ Support  │
│          │  │    └──────────────────────┘    │  │          │
│          │  │                                │  │          │
│          │  └────────────────────────────────┘  │          │
└──────────┘  └────────────────────────────────┘  └──────────┘
```
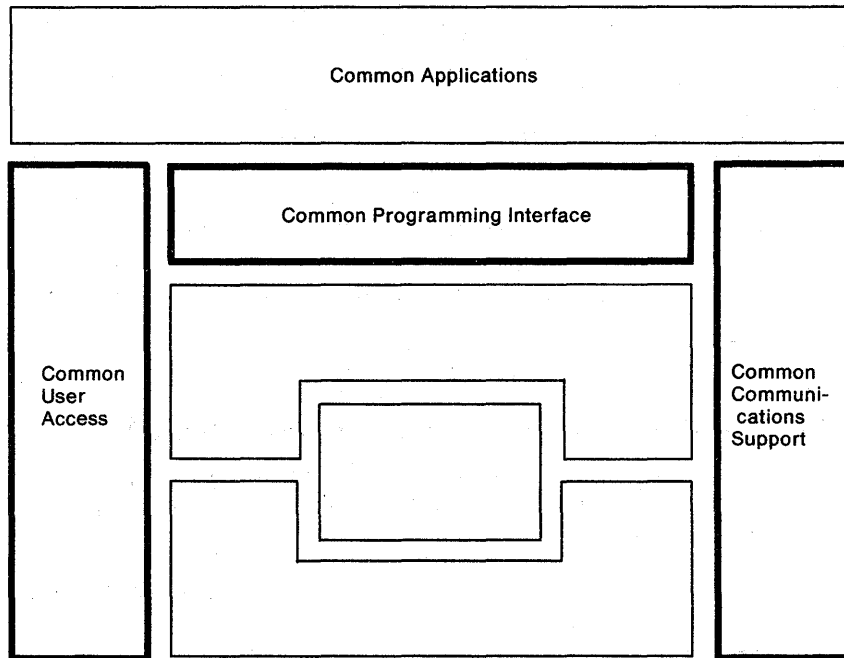
Figure 2. Systems Application Architecture

The *Common User Access (CUA)* defines conventions for developing consistent end-user interfaces to SAA applications. Applications that are developed using the Common User Access guidelines will allow users to interact with the computer in a consistent manner, regardless of which SAA system they are using.

The *Common Programming Interface (CPI)* defines languages and services that allow programmers to write application programs that can run in all SAA environments with little or no change.

Common Communications Support (CCS) defines architectures and protocols that (1) interconnect SAA systems and devices and (2) allow data to be interchanged among them.

Common applications are application programs that are written using elements of the Common Programming Interface; they also conform to the Common User Access conventions. In some cases, common applications directly access CCS functions. Common applications run in, and provide a consistent user interface in, all SAA environments.

The Purpose of Common Communications Support

Common Communications Support (CCS) provides architectures and protocols that allow standardized communication among devices, application programs, systems, and networks, as shown in Figure 3.
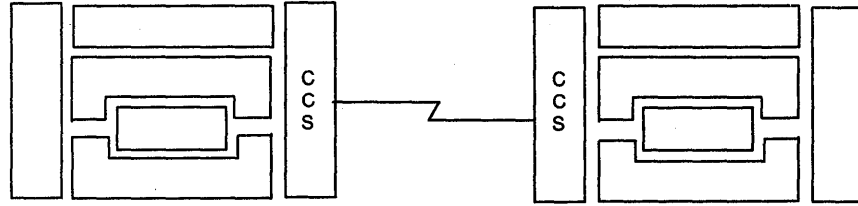
Figure 3. Consistency in Communications

CCS consists of IBM protocols and selected Open Systems Interconnection (OSI) standards that allow both IBM and non-IBM systems to be interconnected. Consistent implementation of CCS architectures will allow networks to be built from systems with vastly differing capacities—from the smallest SAA system to the largest.

Elements of Common Communications Support

Common Communications Support (CCS) comprises elements of Systems Network Architecture (SNA), as well as selected standards from the following international organizations:

- International Telegraph and Telephone Consultative Committee (CCITT) recommendations

- Institute of Electrical and Electronics Engineers (IEEE) standards

- Parts of Open Systems Interconnection.

Elements of CCS are grouped into the six broad categories:

- Objects
- Data streams
- Application services
- Session services
- Network
- Data link control.

The remainder of this chapter introduces CCS categories.

Objects

A person can create a document that contains many types of data (for example, text, graphics, and images); such a document is called a *compound document*. These types of data are contained in *objects*. When documents need to be transferred between SAA systems, they must be in a format and structure that each system can interpret. *Object content architectures (OCAs)* define the structure and content of objects that can exist in a document. The object content architectures currently in Common Communications Support are:

- Presentation Text Object Content Architecture (PTOCA)
- Image Object Content Architecture (IOCA)
- Graphics Object Content Architecture (GOCA)
- Font Object Content Architecture (FOCA).

Objects are transmitted in data streams and can be stored in libraries by applications and hardware components of the network. Chapter 2, "Objects" on page 13 has additional information on object content architectures.

Data Streams

A *data stream* is a continuous ordered stream of data elements conforming to a given format. Application programs can generate data streams destined for a printer, a workstation, or another application program. The data streams in Common Communications Support are:[1]

- Mixed Objects: Document Content Architecture (MO:DCA)
- Intelligent Printer Data Stream (IPDS)
- 3270 Data Stream (3270 DS)
- Revisable-Form-Text: Document Content Architecture (RFT:DCA).

A more detailed discussion of data streams is in Chapter 3, " Data Streams" on page 19.

Application Services

Application services enhance the activity of the network by providing architectures that allow data distribution, document interchange, and network management. The architectures providing application services are:

- **For SNA:**
  - Document Interchange Architecture (DIA)
  - SNA/Distribution Services (SNA/DS)
  - SNA/Management Services (SNA/MS)
  - Distributed Data Management (DDM).
- **For OSI:**
  - File Transfer, Access, and Management (FTAM)
  - X.400 Message Handling System
  - OSI Association Control Service Element (ACSE).

SNA application services are discussed in Chapter 4, " Application Services" on page 27; OSI application services are discussed in Chapter 8, "International Standards" on page 51.

---

[1] The System/3X family will continue to support the 5250 data stream.

Session Services

Session services are required to establish communication between two application programs, to transfer data between the application programs, and to terminate communication between the application programs. The architectures providing session services are:

- **For SNA:**

  - SNA Logical Unit Type 6.2 (LU 6.2) architecture

- **For OSI:**

  - OSI Presentation Layer—Kernel and ASN.1

  - OSI Session Layer—Versions 1 and 2

  - OSI Transport Layer—Classes 0, 2, and 4.

OSI session connection and an SNA LU 6.2 conversation have many similar functions, although the terminology differs. OSI session connection provides the means for associated service users in different networks to organize and synchronize their dialog and to manage their data exchange. An LU 6.2 conversation provides a logical interface through which transaction programs can access the SNA network and its resources. A more complete description of transaction programs is in Chapter 5, "Session Services" on page 35. Additional information on the OSI standards is in Chapter 8, "International Standards" on page 51.

Network

Network services allow connectivity between systems. The architectures that provide network services are:

- **For SNA:**

  - SNA Type 2.1 Node architecture provides low-entry networking

- **For OSI:**

  - Connectionless-Mode Network Services (CLNS) using Internet

  - Connection-Oriented Network Services (CONS) using Subnetwork Interface to X.25.

Type 2.1 Node architecture is discussed in Chapter 6, "Network" on page 39. CLNS and CONS are discussed under "Network Layer" on page 57.

Data Link Control

A *link* consists of transmission media and a data link control protocol. The transmission media may include any combination of telephone lines, microwave beams, fiber optics, satellite links, or coaxial cables. A *data link control* protocol specifies how to interpret control data and how to transmit data across a link.

SAA systems may be interconnected using local area networks, telecommunication links, or packet-switched networks. The primary objective of a local area network is to provide high-speed data transfer among a group of nodes within a building or a group of buildings in a campus or office-complex environment. For SAA systems spread over a wide geographic area, an enterprise requires telecommunication links. In some cases, an enterprise needs to interconnect non-SAA systems to a network of SAA systems. The data-link-control architectures in CCS are:

- *For SNA:*

  - Synchronous Data Link Control (SDLC) to interconnect SAA systems over a wide geographic area.

  - IBM Token-Ring Architecture[2] for local area networks.

- *For SNA and OSI:*

  - X.25[3] to provide access to packet-switched data networks.

Each of these protocols is described in Chapter 7, "Data Link Controls" on page 41.

Note: SNA and OSI have layered structures that are similar, but not identical. In SNA, X.25 is in the Data Link Control layer. In OSI, portions of X.25 are in the Network, Link, and Physical Layers. OSI layers are discussed in Chapter 8, "International Standards" on page 51; additional information on SNA layers is in *Systems Network Architecture: Concepts and Products,* GC30-3072.

CCS Relationships

Application programmers, using elements defined by the Common Programming Interface, can write SAA application programs that interact with data or devices on the same or on different systems; for example:

- Retrieving data from, or storing data in, a file
- Displaying data at a workstation
- Sending documents to a printer.

When any of these activities occurs between SAA systems, the Common Programming Interface (or the application program itself) requires services provided by Common Communications Support.

An Example of How CCS Provides Services for SAA Application Programs

The following example shows services that Common Communications Support can provide for systems in an SAA enterprise.

At an SAA system in London, a programmer codes an application program that needs to:

- Retrieve files from an SAA system in New York

- Integrate information from the New York files into a document that will eventually be printed in Toronto.

  1. The application programmer uses a high-level language that implements the Common Programming Interface to code the program that will retrieve the files and print the document.[4]

     - Type 2.1 Node architecture establishes the connection with the New York system.

---

[2] Institute of Electrical and Electronics Engineers (IEEE) standards 802.2 and 802.5.

[3] International Telegraph and Telephone Consultative Committee (CCITT) recommendation X.25.

[4] *SAA: Writing Applications: A Design Guide,* SC26-4351, provides guidance for coding SAA application programs.

- Synchronous Data Link Control (SDLC) controls the physical transmission of data across a satellite communication link.

- Distributed Data Management (DDM) retrieves the files from the New York system using an LU 6.2 conversation between the two systems.

2. The application program formats the retrieved files for printing.

3. The application program puts the other document parts into their respective object-content-architecture (OCA) formats; it uses the Presentation Interface to put the graphics into the graphics-object-content-architecture (GOCA) format.

4. The application program puts the mixed-object document into the Mixed Object: Document Content Architecture (MO:DCA) data stream.

5. The application program creates a Document Interchange Unit (in Document Interchange Architecture (DIA) format) containing the MO:DCA data stream and specifies the Toronto system as the recipient.

6. The London SNA/DS transmits the document over an LU 6.2 conversation with the Toronto system.

7. SNA/DS in the Toronto system delivers the MO:DCA data stream to the Toronto system service.

8. The print service facility puts the document into the IPDS format and sends the document to the printer.

Part 2. IBM Architectures in CCS

Chapter 2. Objects

A compound document can be made up of several different kinds of data; for example, text, graphics, or image. *Object content architectures* describe the structure and content of each type of data that can exist in a compound document. There are two types of objects associated with a document:

- *Data objects* such as text objects, graphics objects, and image objects

- *Resource objects* such as font objects, which are referred to by data objects.

All objects exist as peers and function as equals. All object content architectures (OCAs) are free to define their own formatting functions. For example, the OCA for text data specifies the spacing between lines and the size of the white space appearing between words.

The object content architectures in Common Communication Support are:

- *Presentation Text Object Content Architecture (PTOCA)* describes presentation-text objects in the document.

- *Image Object Content Architecture (IOCA)* describes image objects in the document.

- *Graphics Object Content Architecture (GOCA)* describes graphic objects in the document.

- *Font Object Content Architecture (FOCA)* defines the structure and content of digital fonts used by data objects in a document.

Additional information about object content architectures is in *Architectures for Object Interchange, GG24-3296.* Future publications will describe each object content architecture.

Object Structure

All objects are made up of two structured fields: an object descriptor and object data. The content of individual fields varies, depending on the kind of object.

Objects are designed to be carried by, and become part of, a data stream. Data streams are used to pass documents between application programs or between an application program and a device. The data stream carrying the object provides all external relationships for the object.

Presentation Text Object Content Architecture

After text has been processed (formatted), it is in presentation form—the text is ready to be presented at a printer. A *presentation-text object* describes the portion of a text document that has been generated from one of many possible sources such as:

- Output from formatting processes

- Direct generation by processes or application programs

- Transformation from text of different presentation formats.

The presentation-text object space defines the area into which the graphic characters will fit when they are presented. This area has no relationship to the physical media or printed page until the final document is actually created.

Minimum Functions Required for PTOCA

PTOCA functions are divided into a PT1 and a PT2 subset. The PT1 subset includes all of the functions required by the most primitive receiver of presentation-text objects. The PT1 subset is the minimum that must be implemented for receivers of presentation-text objects in CCS.

The PT2 subset includes all of the PT1 subset, plus specialized functions such as underscore, overstrike, superscript, and subscript. Detailed information on PTOCA function subsets will be available in a future publication.

Image Object Content Architecture

Many business applications can benefit by being able to include image data in documents. Image data includes such things as signatures, logos, media articles, and photographs. The *Image Object Content Architecture (IOCA)* defines the characteristics of image data in a device-independent format, thereby allowing image information to be interchanged among different applications and devices.

Image characteristics that can be represented by IOCA are:

- Image size
- Resolution
- Recording algorithm
- Compression algorithm
- Number of bits per pixel[5]
- Identification of look-up table.

Minimum Functions Required for IOCA

IOCA function set 10 (FS10) is required for interchanging images in presentation format using the IPDS and MO:DCA data steams. FS10 represents bi-level images which can be compressed using either:

- IBM Modified Modified READ (MMR) compression algorithm
- CCITT T.6 G4 Facsimile compression algorithm.

Function set 20 (FS20) is required for interchanging images in the library format of the MO:DCA data stream. FS20 can represent up to 24 bits-per-pixel color images. Detailed information on IOCA function sets will be published in a future publication.

---

[5] A *picture element (pixel or pel)* is the smallest element of a displayable or printable surface that can be independently assigned color and intensity.

Graphics Object Content Architecture

The term *computer graphics* refers to the definition and representation of graphic elements used to build pictures for presentation either on hard-copy devices (such as printers and plotters) or on soft-copy devices (such as vector or raster displays). The *Graphics Object Content Architecture (GOCA)* uses primitives and attributes to define the structure of computer graphics; GOCA also defines operations for manipulating these graphics.

Structure of Graphic Objects

Segments are the basic units from which a picture is constructed; they are uniquely identified, self-contained collections of primitive drawing orders and attributes. Primitives include things such as:

- Lines and relative lines
- Full arcs, partial arcs, and fillets (rounded corners)
- Character strings
- Areas
- Images.

Typically, attributes describe characteristics of primitives; for example:

- Color
- Line attributes such as type (for example, solid) and width
- Character attributes such as precision, angle, character set
- Patterns.

Every segment is either chained or nonchained. A collection of one or more chained segments defines the picture to be drawn. A picture can be subdivided into "subpictures." Nonchained segments typically define "subpictures" that are incorporated into the main picture by being called from another segment.

Minimum Functions Required for GOCA

The minimum function set required for GOCA interchange is the DR/2V0 function set used in a presentation MO:DCA data stream. DR/2V0 orders are matched to the capabilities of some typical output-only displays and some printers. The functions include curved lines, areas and images.

The DR/3V1 function set is required for interchanging graphics pictures in the library format of the MO:DCA data stream. DR/3V1 includes additional functions such as:

- General clipping paths
- Individual primitive attributes
- Extra curve-generating primitives
- Raster operations to support the requirements of sophisticated workstations.

Detailed information on GOCA function sets will be in a future publication.

Font Object Content Architecture

Graphic characters are the visual representation of symbols used in text; they are letters, numerals, punctuations, or any symbols that represent information. A *font* is a set of graphic characters that have a characteristic design, or a font designer's conception of how associated graphic characters should appear. This sentence shows examples of the *italic font*, **bold font**, and SMALL-CAP FONT.

Font Object Content Architecture (FOCA) defines the parameters required to describe digital fonts used by text and graphic editors, document formatters, and presentation devices. FOCA permits product applications, document references, and presentation devices to access font information.

A method of accessing fonts at the system level is described in the *SAA Common Programming Interface: Presentation Reference, SC26-4359.* Using this method, a system can determine which font resources it has available and can obtain specific information from those resources.

A method for accessing fonts at the printer level is described in *Intelligent Printer Data Stream Reference,* S544-3417. See "Intelligent Printer Data Stream" on page 19 for additional information on IPDS.

How Digitized Fonts Are Used

Font resources are made available for text processing by font production, font storage, and font accessing. FOCA provides the common and consistent font information required for text processing. It also supports font production by defining the font attributes and their interdependencies; however, FOCA does not define how that information is to be generated or modified.

FOCA supports font storage and font access by defining the set of font attributes required by the SAA application environments, and by defining a general format for that information. Application programs that implement FOCA must use the defined font-parameter definitions; however, the application programs are free to define their own internal format for that information.

FOCA defines a set of font-referencing parameters, which may be used to specify and describe a font resource. Each implementing product may specify a set of required font resources; the implementing product may also specify the character content of those font resources. The content of font resources is not defined or controlled by FOCA. For consistency when interchanging and presenting documents, all receiving sites, processing application programs, and presentation devices must have access to the same or equivalent font resources.

FOCA supports the presentation process by allowing device-specific techniques of character-shape representation and presentation. FOCA also permits font producers and product implementers to make use of more generic representation techniques.

Minimum Functions Required for FOCA

The *Font Object Content Architecture (FOCA) Reference,* S544-3285, lists the font parameters (or attributes) defined in FOCA that must be supported for various levels of font information interchange. An interchanged font need not contain all of the information specified by one of the attribute lists, but the processing application programs must be able to accept, build, or pass through all of the information

contained in the supported attribute list without information being lost. To ensure that no information is lost during data interchange, the following attribute lists are required:

- Font descriptive parameters
- Font character set parameters
- Font metric parameters
- Character metric parameters
- Character shape parameters
- Code-page parameters.

Chapter 3. Data Streams

A data stream is a continuous ordered stream of data elements conforming to a given format. The data streams that are part of CCS can transfer data between:

* Two application programs

* An application program and a printer

* A workstation and an application program.

The application programs, workstation, and printer can be on the same system or on different systems.

The data streams that are a part of Common Communications Support are:

* The *Intelligent Printer Data Stream (IPDS)* is the system-to-printer data stream for all-points-addressable printing.

* The *3270 Data Stream* is a formatted data stream used to transmit data between an application program and a nonprogrammable workstation or printer.

* The *Mixed Object: Document Content Architecture (MO:DCA)* is the data stream used to transmit objects from an application program to a programmable workstation or to another application program.

* The *Revisable-Form-Text: Document Content Architecture (RFT:DCA)* is the data stream used to transmit revisable-form text between application programs in an office environment.

Intelligent Printer Data Stream

The Intelligent Printer Data Stream (IPDS) is used to send data created by an application program to an all-points-addressable printer. IPDS can carry any of the document objects mentioned earlier; therefore, it is possible to print pages containing a mixture of different data types. Different application programs can create source data (text, graphics, and image) independently of each other. IPDS allows the output of these independent application programs to be merged when printed so that an integrated mixed-object page results.

Because IPDS is independent of the communication protocol, the same data stream can be transmitted to printers that are attached to channels, controllers, local area networks, or any other communication link that allows transparent transmission of data. IPDS transfers all data and commands through self-identifying structured fields that describe the presentation of one or more pages. IPDS must be part of the printing subsystem of each environment in which IPDS data streams will be interchanged.

All printing subsystems have the following elements in common:

* **Application programs** generate the source data to be printed. Some application programs generate text data that previously would have been directed to line printers. Other application programs generate all-points-addressable text or other data types such as image and graphics for IPDS printers.

- **Presentation services** accept source data and transforms it into an Intelligent Printer Data Stream without changing the existing source data. Presentation services also permit the output of line-printer application programs to be enhanced by IPDS capabilities, such as duplexing, overlays (electronic forms), and multiple high-quality fonts.

- **IPDS printers** accept the Intelligent Printer Data Stream. They can attach to several different system or subsystem environments using one or more communication protocols.

IPDS Functional Divisions

The IPDS architecture is divided into several functional areas, called command sets, each representing a major printer capability. A *command set* consists of:

- IPDS commands, including semantics (the relationship of the command symbol to its meaning)

- Syntax (the command structure/format)

- Architecturally valid values for each field in the command.

In addition, the architecture contains a registry of exception-reporting codes for error conditions in each of its command sets and for printer-related failure, fault, or host-notification conditions.

Each command set is further divided into at least one subset of required function and a subset of optional function. Some command sets have more than one subset of required function. For some command sets, there is a data tower that consists of the data carried in the "Write" command of the corresponding IPDS command set.

The command-set design allows IPDS to support a wide range of printer products. Product developers can match command-set implementations to the specific needs of their product. Figure 4 on page 21 illustrates the IPDS command sets that are part of CCS.
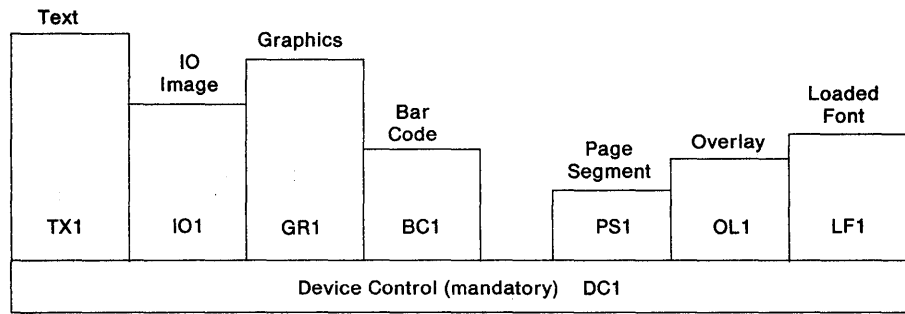
Defined Subsets within IPDS Command Sets (Identified by Command-Set Vectors)
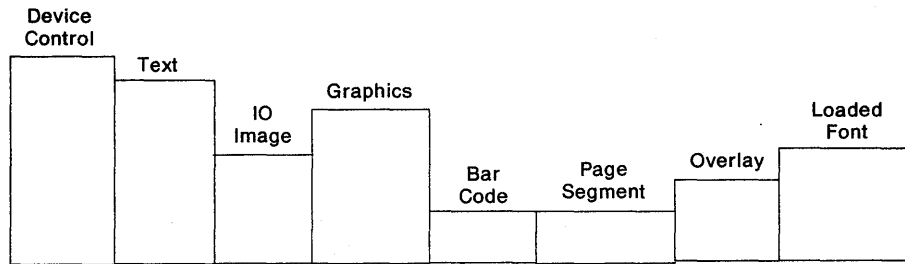
Optional Function within IPDS Command Sets (Identified by Property Pairs)
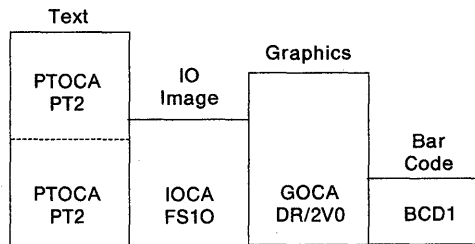
Data Towers

Figure 4. IPDS Command Sets in CCS

Device Control: This command set is composed of the IPDS commands that initialize the environment for a page, communicate device controls, and manage printer acknowledgment protocol.

Text: This command set is composed of the IPDS commands for presenting text information on a page, a page segment, or an overlay.

IO Image: This command set is composed of the IPDS commands for presenting images on a page, a page segment, or an overlay.

Graphics: This command set is composed of the IPDS commands for presenting graphics on a page, a page segment, or an overlay.

Bar Code: This command set is composed of the IPDS commands for presenting machine-readable bar code information on a page, a page segment, or an overlay.

Page Segment: This command set is composed of the IPDS commands to store and present IPDS constructs containing text, graphics, image, and bar code information. These stored constructs, which can be merged with a logical page to assume the current environment, are called page segments.

Overlay: This command set is composed of the IPDS commands to store and present IPDS constructs containing text, graphics, image, and bar code information. These stored constructs are called overlays.

Loaded Font: This command set is composed of the IPDS commands to load and delete font information.

For some IPDS command sets, a data tower exists that consists of the data carried in the "Write" command of the corresponding IPDS command set. A data tower is divided into levels—a higher level of a data tower consists of all lower levels, plus some set of additional functions. Some data tower levels are defined and controlled by other architectures and are simply registered by IPDS:

Text　　This data tower is composed of Presentation Text Object Content Architecture (PTOCA) text controls contained in the data field of the Write Text command. These text controls are required to present text information in a page, a page segment, or an overlay. The text data tower contains two presentation text (PT) levels—PT1 and PT2—defined by PTOCA.

IO Image　This data tower is composed of Image Object Content Architecture (IOCA) self-defining fields contained in the data field of the "Write Image 2" command. These self-defining fields are required to present image data in a page, a page segment, or an overlay. The IO Image data tower contains one level—FS10— defined by IOCA.

Graphics　This data tower is composed of Graphics Object Content Architecture (GOCA) drawing orders contained in the data field of the "Write Graphics" command. These drawing orders are required to present graphics in a page, a page segment, or an overlay. The graphics data tower contains one level—DR/2V0—defined by GOCA.

Bar Code　This data tower is composed of Bar Code data controls contained in the data field of the "Write Bar Code" command. These data controls are required to present machine-readable bar-code information in a page, a page segment, or an overlay. The Bar Code data tower contains one level—BCD1— defined by IPDS.

Minimum Functions Required for IPDS

To claim support of the IPDS architecture, a printer product must do the following:

- Implement the DC1 subset of the device-control command set.
- Implement at least one of the following subsets of the IPDS command sets:
  - Text (TX1)
  - Image (IO1)
  - Graphics (GR1)
  - Bar codes (BC1).

To claim support of the IPDS architecture, a presentation services product must do the following:

- For all commands generated by the presentation services, the command must conform to the IPDS state diagram.

- For all commands generated by the presentation services, the command syntax must conform to the syntax defined by the IPDS architecture.

Command-Set Support Requirements

To claim support of the text, graphics, IO Image, or bar code command sets of IPDS, a printer product must implement an architecturally defined subset of the command set. (Printers can support additional elements of the command set.) In addition, a printer product must also implement a level of the corresponding data tower.

To claim support of any other IPDS command set, a printer product must implement an architecturally defined subset of the command set. (Printers can support additional elements of the command set.)

Data Tower Support Requirements

To claim support of a data tower, a printer product must implement an architecturally defined level of the data tower. Refer to *Intelligent Printer Data Stream Reference*, S544-3417, for additional information on IPDS command sets.

3270 Data Stream

The 3270 Data Stream is a formatted data stream used to transmit data between an application program and a 3270-type workstation or printer. The 3270 Data Stream is based upon the presence of a mapped character buffer in the 3270 workstation. A fixed one-to-one relationship exists between each character-storage location in the buffer and each character position on the display.

An application program uses one of two methods to communicate with the user at a workstation:

- The application program leaves the display surface unformatted and the user uses it in a free-form manner.

- The application program either completely or partially formats the display surface (arranges it into fields) and the user enters data into the fields.

The 3270 Data Stream allows the application programmer to divide the display surface into one active area and, optionally, one or more reference areas; each area is called a *partition*. The partition that is "active" contains a cursor and is the only partition in which the user can enter data or requests.

Minimum Functions Required for 3270 Data Stream

The 3270 functions required for Common Communications Support are called *extended function base support (EBASE)*. EBASE specifies functions in the following categories:

- Query replies

- Structured fields

- Basic 3270 commands

- Basic 3270 orders

- 3270 controls/special characters.

The *IBM 3270 Data Stream Programmer's Reference* identifies the specific functions required in an SAA 3270 Data Stream.

Mixed Object: Document Content Architecture

The Mixed Object: Document Content Architecture (MO:DCA) data stream is used to interchange objects between applications programs; the programs can be in the same system or in different systems.

The MO:DCA data stream consists of the following components:

- Layout structure defines the way objects should be presented.

- Objects (such as text, graphics, and image) define the pieces of a document.

- Mapping that specifies the relationship between the layout structure and objects.

Because a document's layout structure and objects are separate in the MO:DCA data stream, a change in one does not affect any other.

All functions and data that make up an MO:DCA data stream are contained in logical records called *structured fields*. Related structured fields are grouped into categories and bound by unique "begin" and "end" structured-field delimiters. The sample presentation MO:DCA data stream in Figure 5 shows objects in the data stream and the associated begin and end delimiters.
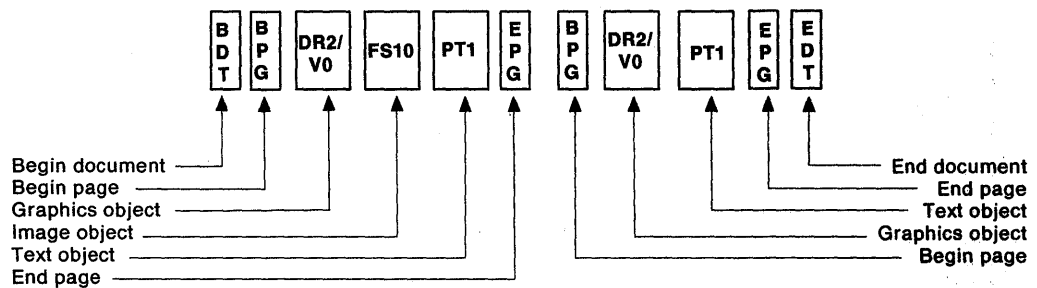
Figure 5. Sample Presentation MO:DCA Data Stream

Minimum Functions Required for MO:DCA

The minimum functions required for interchangeable MO:DCA data streams are based on the intended use of the data stream and are defined by interchange sets. Currently, two interchange sets are defined—one for presentation (MO:DCA-P) and one for library (MO:DCA-L).

An MO:DCA-P data stream is intended for presentation at a workstation or on a printer. MO:DCA-P implementations must support all of the base MO:DCA constructs contained in the interchange set and at least one of the following objects:

- Presentation text (PT1)
- Graphics (DR/2V0)
- Image (FS10).

An MO:DCA-L data stream is intended to save data (usually in a library) for later use by an application program. For example, the Presentation Manager creates an MO:DCA-L data stream known as a *metafile* via calls to its Presentation Interface. MO:DCA-L implementations must support all of the base MO:DCA constructs contained in the interchange set, as well as the following objects:

- Graphics (DR/3V1)
- Image (FS20).

Additional information about MO:DCA can be found in *Architectures for Object Interchange, GG24-3296*. A future publication will have more detailed information on MO:DCA.

Revisable-Form-Text: Document Content Architecture

Revisable-Form-Text: Document Content Architecture (RFT:DCA) defines the structure of a text document that is in a form that can be edited or later formatted. Each recipient of a revisable-form document can modify its contents and format.

An RFT:DCA data stream consists of format units, text units, and an end unit.

- *Format units* contain format declarations and include no text except top and bottom margin text.

- One or more *text units* contain the body of the document.

- The *end unit* identifies the end of the document.

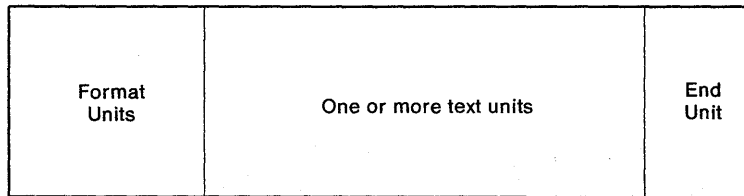| Format Units	One or more text units	End Unit

Figure 6. Major components of an RFT:DCA data stream

Minimum Functions Required for RFT:DCA

The minimum functions required for RFT:DCA interchange are:

- Format unit 1
- Format unit 2
- One or more text units
- An end unit.

A text unit must contain at least one body-text structured field.

Chapter 4. Application Services

Application services enhance the function of the network by providing a means for application programs to distribute files, exchange documents, and exchange electronic messages. The SNA protocols that provide application services are described in this chapter; OSI protocols that provide application services are described in "Application Layer" on page 54.

Document Interchange Architecture

The information handled in today's offices takes many forms—text, graphics, images, data, voice. When speaking of Document Interchange Architecture (DIA), the term "document" refers to a block of information of any size, regardless of its form. This definition includes facsimile images, binary data files, digitized audio files, and other information not usually thought of as documents.

Document Interchange Architecture is a program-to-program communication architecture that allows documents to be interchanged among a broad spectrum of office systems. Users can add many parameters to DIA commands that communicate, without ambiguity, the user's instructions for filing, accessing, retrieving, and revising a document.

The coding scheme of DIA is flexible enough to convey any document, regardless of its content, from one SAA system to another. DIA delivers documents to users on the same system as the originator. When a document needs to go to users on a system different from the originator's system, DIA invokes SNA/Distribution Services (SNA/DS). (SNA/DS is discussed under the topic, "SNA/Distribution Services" on page 29.)

Document Interchange Architecture Services

DIA performs several services on behalf of the end user:

* Document distribution services allow a person or an application program to deliver documents to recipients in the network. A person can schedule a distribution by document priority, confirm its delivery, and obtain reported errors. These services are commonly called electronic document distribution.

* Document library services allow a person or an application program to file documents in an available document library, retrieve them from the library, delete them from the library, or search for particular documents in the library.

* File transfer services allow a person or an application program to file, retrieve, and deliver documents to and from user document libraries.

* Application processing services execute user-created programs, format documents, modify document-description and access controls, and deliver documents.

DIA allows users to restrict the following DIA services:

* Modifying document attributes

* Receiving documents in a distribution

* Acting on behalf of another user.

The Structure of DIA

The basic unit of information exchanged between DIA processes is the *document interchange unit (DIU)*, shown in Figure 7.

Prefix	Command Sequence	Data Units	Document Units	Suffix

Figure 7. Structure of a Document Interchange Unit

- The *prefix* contains information that introduces and identifies the DIU.

- The *command sequence* contains the commands that specify the function to be performed and related processing information. The command sequence contains from 1 to 255 commands.

- The *data unit* contains information that may be referred to by one or more DIA commands in the command sequence. This is an optional field. A DIU can have up to 255 data units.

- A *document unit* contains the document profile and optionally contains the document content. The document unit field is optional; it is only present when a document profile and content are sent from one DIA process to another. A DIU can contain up to 255 document units.

- A *suffix* identifies the end of the DIU and reports whether any abnormal conditions occurred while the DIU was being processed.

Minimum Functions Required for DIA

Figure 8 lists the DIU entities that a product must support for SAA conformance.

Prefix	The Interchange form must be supported.
Command	Required support for one command per DIU.
Data Unit	Support not required.
Document Unit	Support not required.
Suffix	Normal and Abnormal termination forms must be supported.

Figure 8. DIU Entities Required for SAA

Figure 9 on page 29 lists the DIA function sets that make up the DIA SAA subset. Function set 10 is required for all DIA products. The other function sets are required for products providing the associated service.

DIA Service	Function Set	Comments
Session Services	10	Required Support
Document Library Services	8	
Document Distribution Services	2 5	Distribution Server Distribution Requester
Application Services	9	

Figure 9. DIA Services in SAA

Refer to *Document Interchange Architecture Technical Reference,* SC23-0781, for additional information on DIU entities and DIA services.

SNA/Distribution Services

SNA/Distribution Services (SNA/DS) provides a general-purpose, connectionless communications service to application programs that use it. With a *connectionless* service, communication occurs without a direct connection being established between the communicating application programs; connectionless service is also commonly known as a *messaging* service. In contrast, a *connection-oriented* service is one that does provide a direct connection between the communicating application programs. An LU 6.2 conversation between two application programs is an example of a direct connection between the programs.

SNA/DS allows application programs to communicate without requiring that the origin and destination of the communications both be active simultaneously. The architecture allows the nodes at which the origin and destination application programs reside (not the application programs themselves) to communicate via direct sessions. The origin and destination nodes may communicate via intermediate nodes that provide a store-and-forward function. Traffic is queued, if necessary, before being sent from one node to the next.

The connectionless nature of the service does not necessarily imply long delays in completing the processing of requests. Because the communicating application programs do not interact, the responsibility for processing requests shifts from the originating program to SNA/DS, which subsequently transfers the request to the destination application program. After the distribution service has accepted the request, it is independently responsible for carrying it out.

The SNA/DS Network

Figure 10 on page 30 shows that distribution services consist of a network of nodes known as *distribution service units* (DSUs). Distribution service units communicate with one another via LU 6.2 conversations.
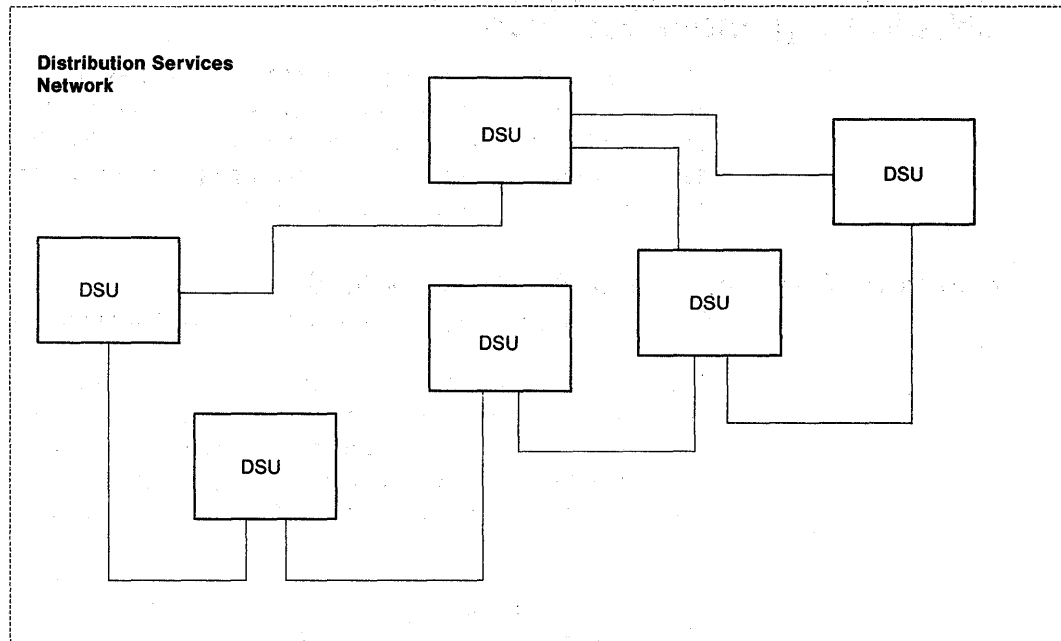
Figure 10. An SNA/Distribution Services Network

The unit of work performed by the distribution service is a *distribution*; a distribution begins at one DSU and may spread out to many. The work performed on a distribution includes the acceptance of the request at the origin, the generation and movement of copies of the distributed material across the network, and the delivery of those copies to the specified destinations. Various levels of service may be requested for distributions; for example, higher or lower priorities.

Minimum Functions Required for SNA/DS

Because the SNA/DS architecture can be implemented in equipment of different sizes to serve various application purposes, it is not unusual for a given implementation to choose to implement less than the complete architecture. The SNA/DS architecture defines the rules under which implementations may choose to omit functions. Implementations make choices in six categories:

- Protocol boundary exposure
- Role
- Base and option sets
- Electives
- Specializations
- Optimizations (no observable function omitted).

The effect of these choices upon the total size of the implementation can be significant. For a further description of the SNA/DS architecture and the alternatives available to implementations, refer to the *SNA/Distribution Services Reference*, SC30-3098.

SNA/Management Services

SNA/Management Services (SNA/MS)[6] enable IBM and non-IBM products to monitor and control networks in a consistent fashion. *Management services* allow users to plan, organize, and control a data communication network. SNA/MS defines formats and protocols for managing problems, accounting and performance, and the configuration in a network.

Minimum Functions Required for SNA/MS

Currently, only problem management is part of Common Communications Support.

SNA/MS provides problem management by establishing focal points and entry points in a network. A problem-determination *focal point* is the system in the network in which problem-determination functions are centralized for several systems in the network. A network *entry point* gathers network-management data for itself and resources attached to it; the entry point forwards the data to its focal point.

The number and placement of focal points is determined by the enterprise's requirement for central- versus distributed-management of the network. A centrally managed network may have one operator console for a single network. Complex networks with diverse groupings of resources may require multiple operators, thereby distributing control of the network.

Problem management allows a focal point to determine errors or failures in the network. When a problem occurs, an entry point notifies its focal point by sending an *alert*.

Additional information about SNA/MS problem management can be found in the publication, *SNA Management Services Reference, SC30-3346.*

Distributed Data Management

The Distributed Data Management (DDM) architecture allows files to be shared among SAA systems. It contains a data connectivity language that allows data interchange among different kinds of systems. DDM has a vocabulary of terms and a set of rules (a grammar) for using the terms. DDM also contains a set of standardized file models and access methods.

DDM architecture contains functional descriptions and the line protocols to express these functions; it also contains models of the managers and programs that support the functions. This logical modeling of an operating system's data management is the heart of the interoperability enabled by the DDM architecture.

---

6 SNA/Management Services was originally announced in Systems Application Architecture as Network Management Architecture.

Minimum Functions Required for DDM

The DDM functions required for Common Communications Support are listed below. Two terms used in describing the required subset are "source" and "target."

Source The end of the communication link at which a request originates; for instance, the source agent, the source system, the source security manager.

Target The end of the communication link where the requested user data or file server resides; for instance, the target agent, the target system, the target security manager.

Summary of Required Functions for Source and Target DDMs

Each system that provides source or target support must implement the following managers:

- Communications manager for SNA LU 6.2
- Source or target agent
- Supervisor
- Lock manager
- Security manager
- File manager for direct files
- File manager for keyed files
- File manager for sequential files
- Record manager for fixed-length records
- Access-method managers for the following types of access:
  - Relative by record number—sequential and direct
  - Random by record number—direct
  - Combined by record number—direct
  - Relative by key—keyed
  - Random by key—keyed
  - Combined by key—keyed.

Summary of Optional Functions for Source and Target DDMs

The following managers are optional for the SAA DDM source or target.

- File manager for alternate index files
- File manager for directories
- File manager for stream files
- Record manager for initially varying-length records
- Record manager for varying-length records
- Access-method managers for the following types of access:
  - Combined access—sequential, direct, and keyed
  - Random by record number—direct
  - Combined by record number—direct
  - Directory entry
  - Byte stream.

Summary of the DDM Commands

The DDM architecture contains commands to work with files at the file level and at the record level. Within the DDM definition for Common Communications Support, the commands are specified as "required" or "optional." Refer to the *DDM Architecture Level 2.0 Reference Manual, SC21-9526,* for a complete description of SAA Common Communications Support DDM architecture definition and additional information about the commands and the managers.

Chapter 5. Session Services

SNA Logical Unit Type 6.2 (LU 6.2) architecture provides session services for Common Communications Support. LU 6.2 defines the formats and protocols for general-purpose program-to-program communication. (Whenever the term logical unit (LU) is used in this chapter, the reference is to LU 6.2.)

How Communication Occurs Using LU 6.2

A *transaction program* processes transactions in a network. One example of a transaction is the entry of a customer's deposit that results in the customer's balance being updated. Another example is the process of recording item sales, verifying checks before accepting them as tender, and receiving payment for the items sold. A third example is the transfer of a message to one or more destinations in the network.

When a transaction program needs to communicate with another transaction program, its associated logical unit establishes an LU-LU session with the other transaction program's associated logical unit. After the LU-LU session is established, the two LUs provide a connection between their transaction programs. Communication between two transaction programs is called a *conversation*. Conversations use a session serially and thus form an efficient method for sharing the communication path among different pairs of transaction programs over time, without the overhead of initiating a new session for each transaction. Two kinds of transaction programs can participate in LU 6.2 conversations—application transaction programs and service transaction programs

Application transaction programs access the network through LU 6.2 and process transactions cooperatively with other application transaction programs in the network. Application programmers can use the Communications Interface when they need their program to communicate directly with another application transaction program. The Communications Interface is a collection of high-level language statements that generate the LU 6.2 formats and protocols. Refer to *Common Programming Interface: Communications Reference, SC26-4399,* for a description of the interface.

Service transaction programs are IBM-supplied programs that the architecture defines. They execute within LU 6.2, and typically provide utility services to application transaction programs. For example, some service transaction programs provide document interchange services (using DIA) that allow processors and workstations to exchange documents synchronously. Other service transaction programs provide SNA/Distribution Services (SNA/DS) that allow asynchronous distribution of files and documents. Still other service transaction programs provide Distributed Data Management (DDM) functions that allow application transaction programs to share access to data files stored anywhere in the network.

In Figure 11 on page 36, program A is in conversation with program C, and program B is in conversation with program D. Program A could use the session between LU X and LU Z to initiate a conversation with program D. Likewise, program B could use the session between LU X and LU Y to initiate a conversation with program C.
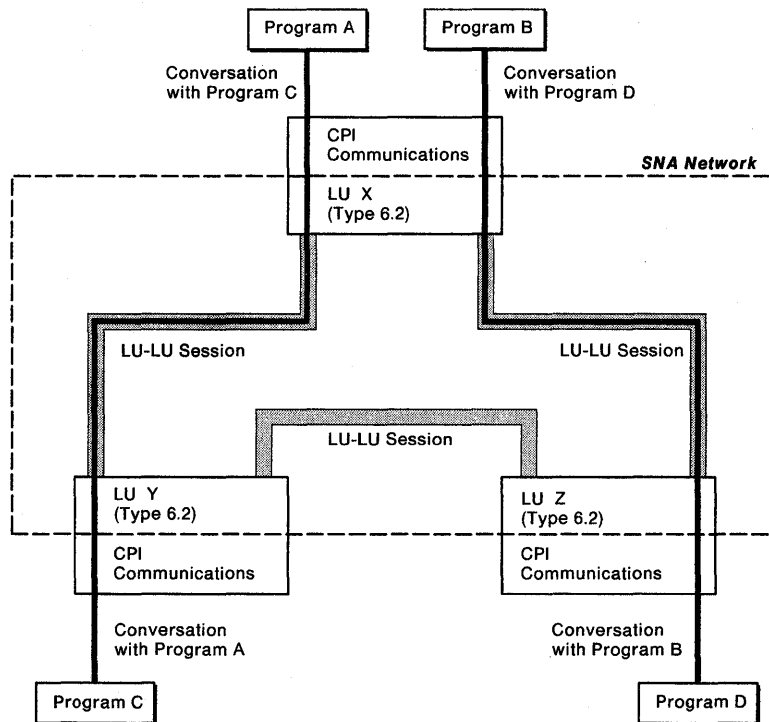
Figure 11. Logical Unit 6.2 Conversations

Minimum Functions Required for LU 6.2

The LU 6.2 functions are classified into a base set (mandatory) and various option sets. Implementation of the option sets is a product or application choice, but only complete option sets can be implemented. If a higher-level option set is implemented, then all prerequisite (underlying or supporting) option sets must also be implemented.

The base is extensive enough to support meaningful connectivity between any two products, yet basic enough to allow nonprogrammable devices to be universal among product implementations. Consequently, any two products implementing LU 6.2 should be able to communicate at the base level, and in fact at the highest common function level, assuming of course, that a compatible transaction program is located at each end.

Common Communications Support for LU 6.2 consists of the base set. In addition, the following option sets are required to support the Common Programming Interface for Communications.

101 **Flush the LU's send buffer:** a program can explicitly cause the LU to flush its send buffer.

102 **Get attributes:** a program can obtain attributes of a mapped conversation.

105 **Prepare to receive:** a program can change the conversation from send state to receive state and at the same time flush the LU's send buffer, request confirmation, or request synchronization point.

106   **Receive immediate:** a program can receive whatever information is available on a conversation without having to request posting of the conversation.

110   **Get conversation type:** a program that supports both the basic conversation and mapped conversation protocol boundaries can determine which category of verbs it should use in conjunction with a resource ID.

203   **Immediate allocation of a session:** a program can allocate a contention-winner session only if one is immediately available; otherwise, the allocation is unsuccessful.

245   **Test for request-to-send received:** a program can test to determine whether a request-to-receive notification has been received on a conversation; for example, following synchronization point processing.

290   **Logging of data in a system log:** a program can record error information in the system's error log.

Additional information on LU 6.2, its base set, and option sets can be found in the following manuals:

- *SNA Transaction Programmer's Reference Manual for LU Type 6.2*, GC30-3084

- *SNA LU 6.2 Reference: Peer Protocols*, SC31-6808.
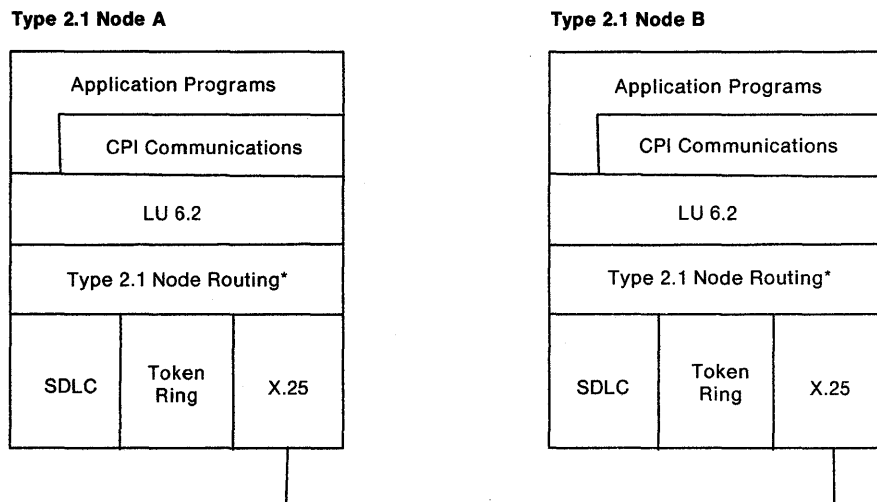
Chapter 6. Network

IBM's Type 2.1 Node architecture, which provides low-entry networking, allows direct connectivity between two SAA systems. The two systems can establish session connections on a peer-to-peer basis. *Peer-to-peer* means that either system is able to activate sessions.

Establishing Connectivity between Systems

A type 2.1 node can establish sessions with adjacent type 2.1 nodes. With Type 2.1 Node architecture, SAA systems can be connected in a peer-to-peer fashion—two programmable workstations, two AS/400s, a programmable workstation and an AS/400.

Enhanced connectivity between SAA systems is further aided by Type 2.1 Node architecture's support of CCS data link controls—X.25, IBM Token-Ring, and Synchronous Data Link Control (SDLC).

Figure 12 shows SAA application programs using the Type 2.1 Node architecture to communicate in an SAA environment. The programs make calls to the CPI for Communications in order to exchange data with the partner program in the other system. The application program uses the Communications Interface to access the services provided by LU 6.2. LU 6.2, in turn, uses Type 2.1 Node architecture as the routing connection to the partner node.

Type 2.1 Node A　　　　　　　　　　　　　**Type 2.1 Node B**

Application Programs
CPI Communications
LU 6.2
Type 2.1 Node Routing*

SDLC	Token Ring	X.25

Application Programs
CPI Communications
LU 6.2
Type 2.1 Node Routing*

SDLC	Token Ring	X.25

* Includes Path Control and Control Point Logic.

Figure 12. Type 2.1 Node Architecture in Systems Application Architecture

Minimum Functions Required for Type 2.1 Node Architecture

The following functions of Type 2.1 Node architecture are a part of Common Communications Support.

- Peer-to-peer protocols allowing either side to initiate link activation and deactivation

- Peer-to-peer protocols allowing either side to initiate session activation and deactivation

- Support of XID3 protocols allowing link station role-negotiation capability

- Support of XID3 protocols allowing use of Exchange State indicators to govern XID exchange

- Supported data link controls: SDLC, Token-Ring, and X.25

- Support of switched and nonswitched link connections

- Support of independent LU-LU sessions.

The Type 2.1 Node architecture is described in *SNA Type 2.1 Node Reference*, SC30-3422.

Chapter 7. Data Link Controls

Several terms are widely used when discussing data link controls. *Data terminal equipment (DTE)* is the standard term used for a communication device that sends or receives network data. DTEs can be computers, cluster controllers, or workstations. DTEs are connected to *data circuit-terminating equipment (DCE)*, which in turn connects the user's equipment to a data network. The function of the DCE is separate from that of the DTE, but the DCE may be part of the same physical package as the DTE.   Figure 13 shows the components of a data link.

Each node that communicates with another node over transmission media requires a link station and a DCE. A *link station* is the hardware and software that allows a node to attach to, and provide control, for a link. The link station is part of the DTE. The part of the data link that includes the DCEs and the channel between them, but not the link stations, is called the *link connection*. (Another term for link connection is *data circuit*.)

DCE   Data Circuit-terminating Equipment
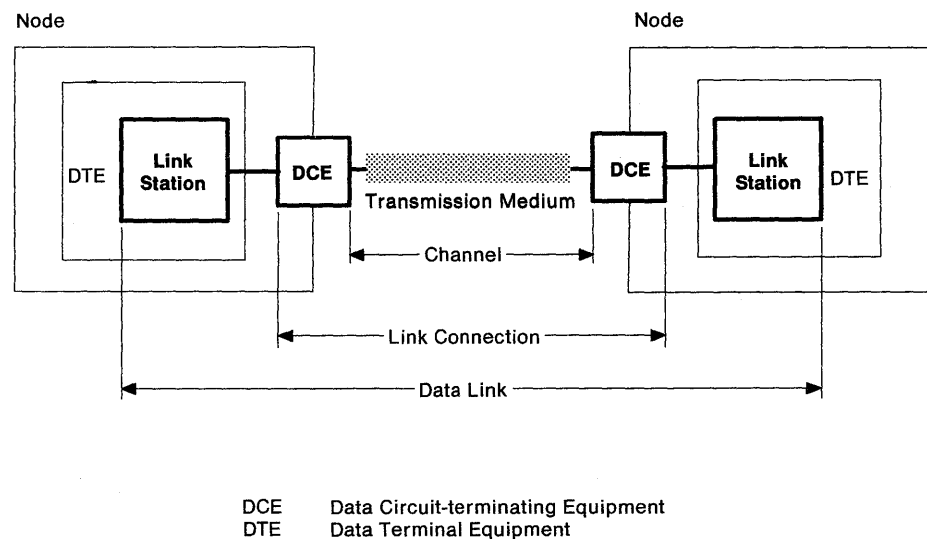DTE   Data Terminal Equipment
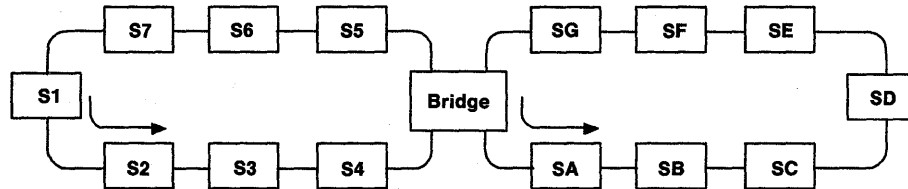
Figure 13. Components of a Data Link

Frame is another term commonly used when talking about data link controls. A *frame* is the basic unit of information transmitted over a data link. It includes whatever is required by the data link control; for example, delimiters, controls, data, data-checking characters.

Common Communications Support includes data link controls that allow data to be transmitted in local-area networks, packet-switched networks, and wide-area networks.

Token-Ring Architecture

The IBM Token-Ring architecture provides data link control for local area networks; it is based on the IEEE 802.2 and 802.5 standards, and the ISO 8802-2 and 8802-5 standards.

In a token-ring network, a ring consists of ring stations and the transmission medium to which they are attached. A ring station contains the combination of functions that allows a DTE to attach to the ring and to use the access protocols. Each ring station can serve one or more attached DTEs, allowing them to communicate with other attached devices on the ring. A ring network configuration consists of a series of ring stations connected by unidirectional transmission links to form a closed path. Figure 14 shows an example of two token-ring networks connected by a bridge. The actual addresses of the stations are independent of their position in the ring.

```
   ┌──[S7]──[S6]──[S5]──┐      ┌──[SG]──[SF]──[SE]──┐
[S1]                    └[Bridge]                    [SD]
   └──[S2]──[S3]──[S4]──┘      └──[SA]──[SB]──[SC]──┘
```

- S1 through S7 and SA through SG are ring stations.

- The bridge connects two token rings.

Figure 14. Multiple-Ring Connections.

Accessing the Token-Ring Network

A unique bit sequence, called a *token*, is passed from one ring station to another along the path indicated by the arrow in Figure 14. If a ring station receives a token and has data to send, the station can initiate data transmission in a frame by including the address of the recipient. As soon as the station completes its transmission, it releases a new token. Meanwhile, the transmitted token passes to each ring station; each station regenerates the frame and passes it on to the next ring station. The intended recipient retains a copy of the frame and sets a response bit in the frame. When the frame gets back to the originating ring station, the transmission is complete and a token is passed to the next node on the ring.

A token-ring link consists of two layers, one of which is divided into two sublayers:

- The *physical layer*[7] controls the physical attachment of stations to the ring and the encoding of data bits into electrical signals.

- The *data link layer* provides the functions and procedures used to establish, maintain, and release data link connections between elements of the network.

  - The Medium Access Control (MAC) sublayer[7]

    - Controls the sharing of the common transmission medium by all the attached stations, and controls the routing of information between the physical transmission medium and the Logical Link Control sublayer. It defines formats and protocols for control of the ring and the exchange of user data.

---

[7] IEEE standard 802.5 specifies a ring using a token-passing scheme for access.

    — The Logical Link Control (LLC) sublayer[8]

        — Defines formats and protocols for exchanging frames between LLC sublayers attached to a local-area network. It has provisions that ensure that error-free, nonduplicated, properly ordered frames are delivered to the appropriate data-link user.

Minimum Functions for Token-Ring

The minimum functions for the Token-Ring architecture are:

- Dynamic window flow control in LLC

- Source routing for bridging token rings

- The MAC layer enhancement for Token-Ring architecture

- Early token release for 16 Mbps operation of the token ring.

Additional Information

The following manuals contain additional information on the IBM Token-Ring:

- *Introduction to Local Area Networks, GC20-8203*

- *IBM Token-Ring Network Introduction and Planning Guide, GA27-3677*

- *IBM Token-Ring Network Architecture Reference, SC30-3374.*

X.25 in Packet-Switched Data Networks

In a *packet-switched data network (PSDN)*, users do not have exclusive right to a specific physical circuit. Instead, many network users share the same circuits for transmitting their messages. Messages are divided into segments called *packets*; packets are transmitted independently through the network until they reach their destination node. If a particular circuit is too crowded or not working, packets may be routed over a different circuit. At the destination node, the packets are reassembled into their original order.

Figure 15 on page 44 shows how physical circuits are shared by many DTEs in a packet-switched data network.

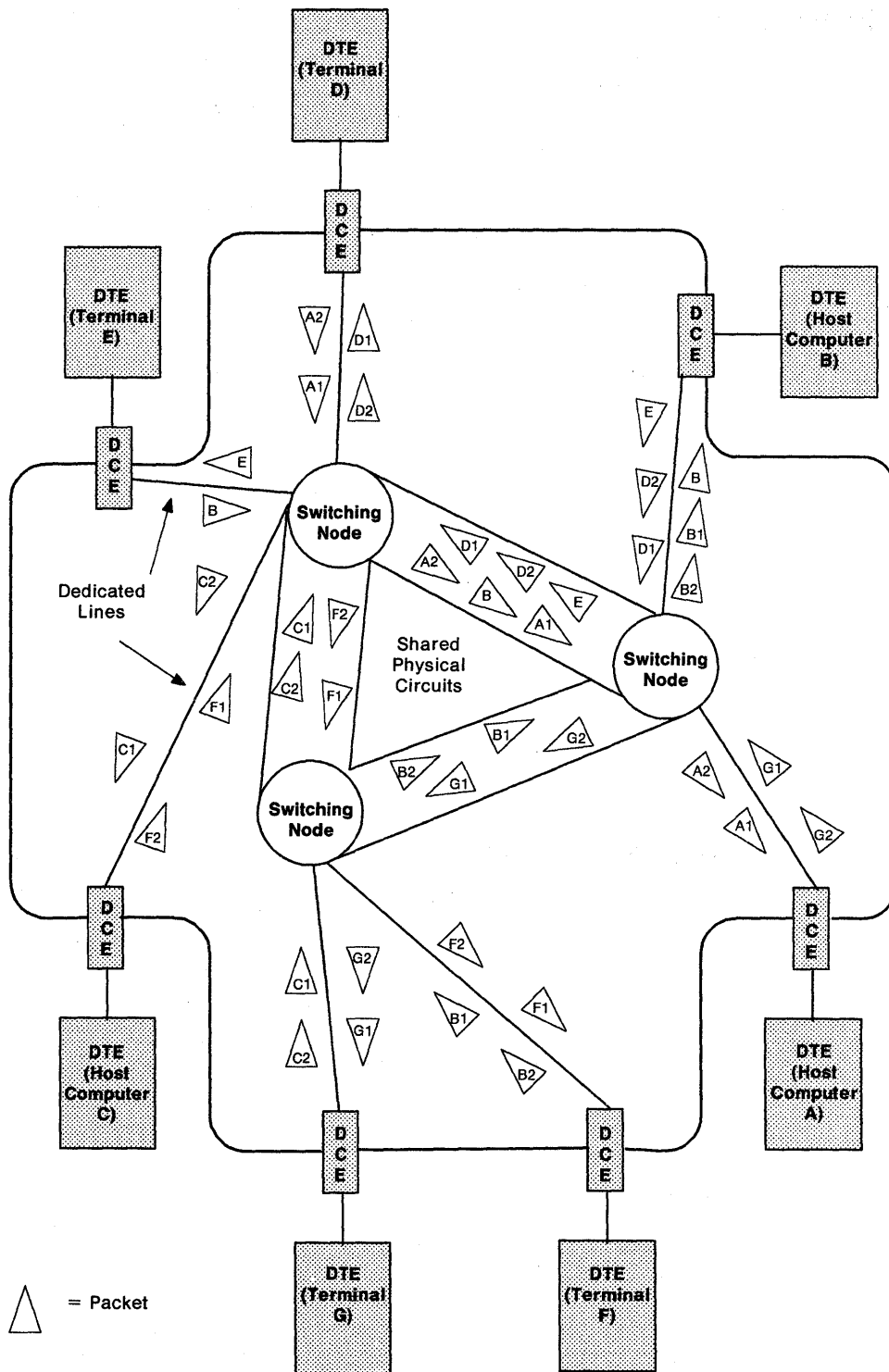[8] IEEE standard 802.2 specifies Logical Link Control (LLC).

Figure 15. A Packet-Switched Data Network

X.25 Frames

In the X.25 environment, packets are transmitted in frames through the link. A *frame* is the link-level vehicle for transmitting commands and responses over the physical circuit between a DTE and a DCE. The three types of frames are supervisory, unnumbered, and information. Supervisory and unnumbered frames carry only link-control information. Information frames carry one packet of data or one packet of control information over the DTE/DCE circuit. Figure 16 shows a frame and a packet.
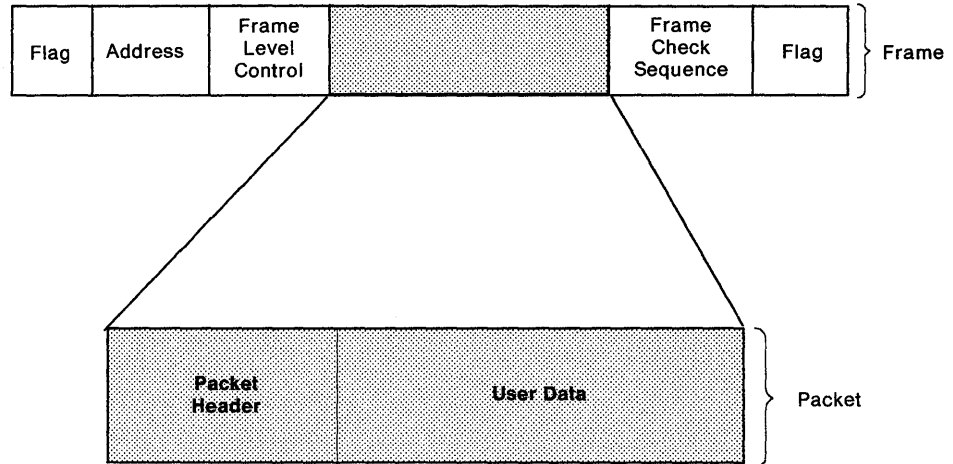
Figure 16. X.25 Frame and Packet

Minimum Functions Required for X.25

The minimum functions required for X.25 in CCS is the implementation of SNA-to-SNA connections or OSI-to-OSI connections. Refer to "Network Layer" on page 57 for information on X.25 in Open Systems Interconnection. SNA-to-SNA connections allow a pair of IBM SNA X.25 (1984) DTEs to be connected via virtual call, permanent virtual circuit services, or both. More detailed information on SNA-to-SNA connections is in *The X.25 1984 Interface for Attaching SNA Nodes to Packet-Switched Data Networks—General Information, GA27-3345* and *The X.25 1984 Interface for Attaching SNA Nodes to Packet-Switched Data Networks—Architecture Reference, SC30-3409.*

Additional Information

IBM's interface to X.25 protocols is documented in the following manuals:

* *The X.25 Interface for Attaching SNA Nodes to Packet-Switched Data Networks—General Information, GA27-3761*

* *The X.25 1984 Interface for Attaching SNA Nodes to Packet-Switched Data Networks—General Information, GA27-3345*

* *The X.25 1984 Interface for Attaching SNA Nodes to Packet-Switched Data Networks—Architecture Reference, SC30-3409*

In addition, CCITT Recommendation X.25 is documented in:

- *CCITT Yellow Book Volume VIII—Fascicle VIII.2 Recommendations X.1 − X.29 (Geneva 1980)*

- *CCITT Red Book Volume VIII—Fascicle VII.3 Recommendation X.20 − X.32 (Malaga-Torremolinos 1984)*

Synchronous Data Link Control

Synchronous Data Link Control (SDLC) is a discipline for managing synchronous, code-transparent, serial-by-bit information transfer between nodes connected by data links.[9] Data may be sent simultaneously in both directions (referred to as *two-way simultaneous* transmission) or alternately, in one direction at a time (referred to as *two-way alternate* transmission.)

The link connection may have point-to-point or multipoint configuration; a point-to-point link may be nonswitched or switched. SDLC includes comprehensive detection and recovery procedures for transmission errors that may be introduced onto the link.

SDLC Link Connections

An SDLC link connection can have one of the basic configurations shown in Figure 17 on page 47.

- Nonswitched point-to-point

- Switched point-to-point

- Nonswitched multipoint.

In a nonswitched configuration, the link connection exists for a period of time, independently of whether it is being used to transmit data. If the link connection is contracted-for rather than owned, the channels and transmission media used by the link connection may vary from time to time.

In a switched configuration, a connection is established each time there is data to be transmitted, and the connection is broken after transmission is completed. Each time a switched connection is established, it is likely to use a different combination of channels and transmission media.

A point-to-point configuration has two link stations; a multipoint configuration has three or more link stations. One link station is called the *primary link station*; it controls use of the link by all the link stations attached to it. The rest of the link stations on the link are *secondary link stations*. In a multipoint configuration, the secondary link stations communicate only with the primary link station—never with each other.

---

[9] Synchronous Data Link Control (SDLC) is a proper subset (normal response mode) of the ISO High-Level Data Link Control (HDLC) defined by the ISO Standards listed under "Additional Information" on page 48.

Nonswitched Point-to-Point

Switched Point-to-Point

Nonswitched Multipoint

DCE — Data Circuit-terminating Equipment
DSE — Data Switching Exchange
LS — Link Station
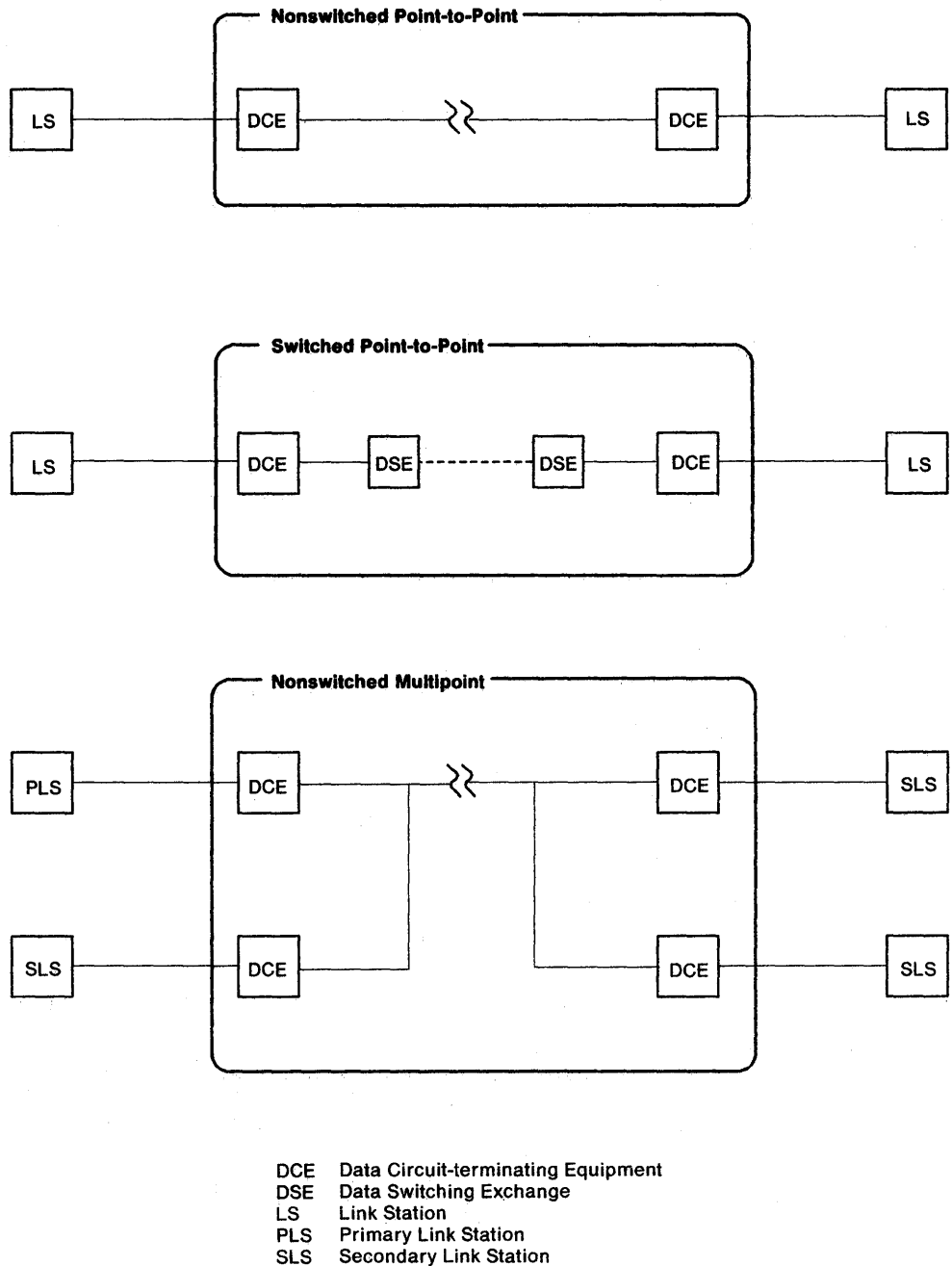PLS — Primary Link Station
SLS — Secondary Link Station

Figure 17. Link Connection Configurations

Data and control signals can flow in either direction over the link connection. Whether they can flow simultaneously in both directions, or in only one direction at a time, depends on the equipment. The term *duplex* refers to the capability of the channel and the link connection to transfer data in both directions at once. The term *half-duplex* refers to the capability of the channel and the link connection to transfer data in both directions, but *not* at the same time.

The qualifier *duplex* and *half-duplex* can be applied to the configurations mentioned earlier. The possible configurations are:

• Half-duplex, nonswitched point-to-point
• Duplex, nonswitched point-to-point

- Half-duplex, switched point-to-point
- Duplex, switched point-to-point
- Half-duplex, nonswitched multipoint
- Duplex, nonswitched multipoint.

Minimum Functions for SDLC

There are no minimum functions because SDLC cannot be implemented using a subset of the architecture.

Additional Information

Synchronous Data Link Control is described in *Synchronous Data Link Control Concepts*, GA27-3093.

High-Level Data Link Control is described in the following international standards:

- *International Standard Data Communications—High-Level Data Link Control Procedures – Frame Structure*, ISO 3309

- *International Standard Data Communications—High-Level Data Link Control Elements of Procedures*, ISO 4335

- *International Standard Data Communications—High-Level Data Link Control Procedures – Consolidation of Classes of Procedures*, ISO 7809.

Part 3. International Standards in CCS

Chapter 8. International Standards

The International Standards Organization (ISO) has identified the requirements for interconnecting systems that contain hardware and software built by different manufacturers. ISO developed a set of international standards that define a reference model, protocols, and service primitives for open systems communication. This set of standards is known as *Open Systems Interconnection (OSI)*. Systems that adhere to OSI standards are said to be "open" to one another and are called *open systems*.

The ISO reference model shown in Figure 18 on page 52 identifies seven layers and specifies the services each layer provides to, and receives from, adjacent layers. Each layer communicates with its equivalent layer in another open system by using the protocols defined by OSI standards.
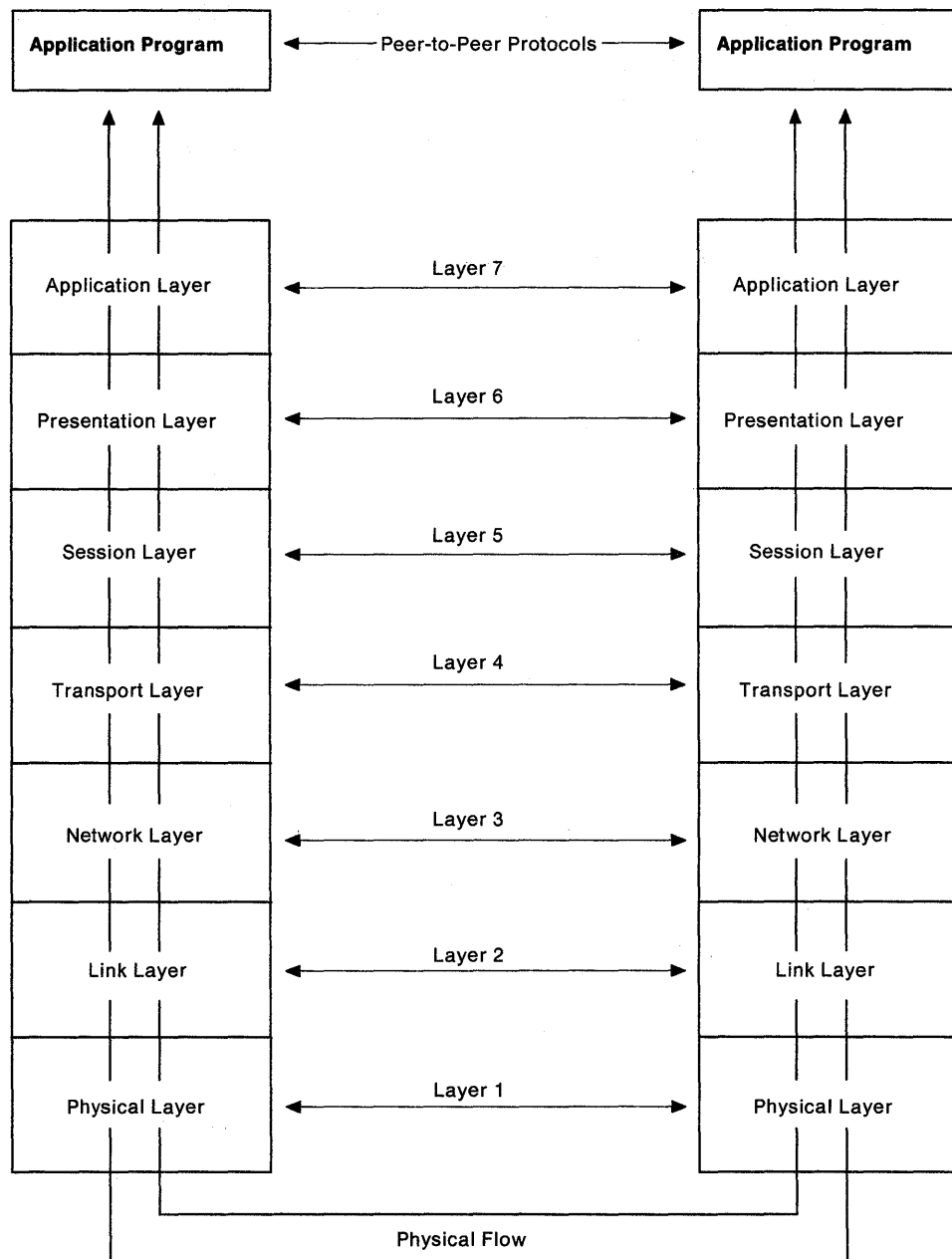
Figure 18. ISO Reference Model

The following OSI standards are supported in Common Communications Support:

- File Transfer, Access, and Management (FTAM)

- X.400 Message Handling System

- Association Control Service Element (ACSE)

- Presentation Layer—Kernel and Abstract Syntax Notation Number 1 (ASN.1)

- Session Layer—Versions 1 and 2

- Transport Layer—Classes 0, 2, and 4

- Network Layer—Connectionless Network Services (CLNS)

- Network Layer—Connection-Oriented Network Services (CONS).

Figure 19 shows these standards in the ISO reference model.

Application Layer FTAM (ISO 8571/1-4) X.400 MHS ACSE (ISO 8650)	**Layer 7**
Presentation Layer Kernel (ISO 8823) ASN.1 (ISO 8825)	**Layer 6**
Session Layer Versions 1 and 2 (ISO 8327)	**Layer 5**
Transport Layer Classes 0, 2, and 4 (ISO 8073)	**Layer 4**
Network Layer CLNS (ISO 8473) CONS (ISO 8878)	**Layer 3**
Link Layer	**Layer 2**
Physical Layer	**Layer 1**
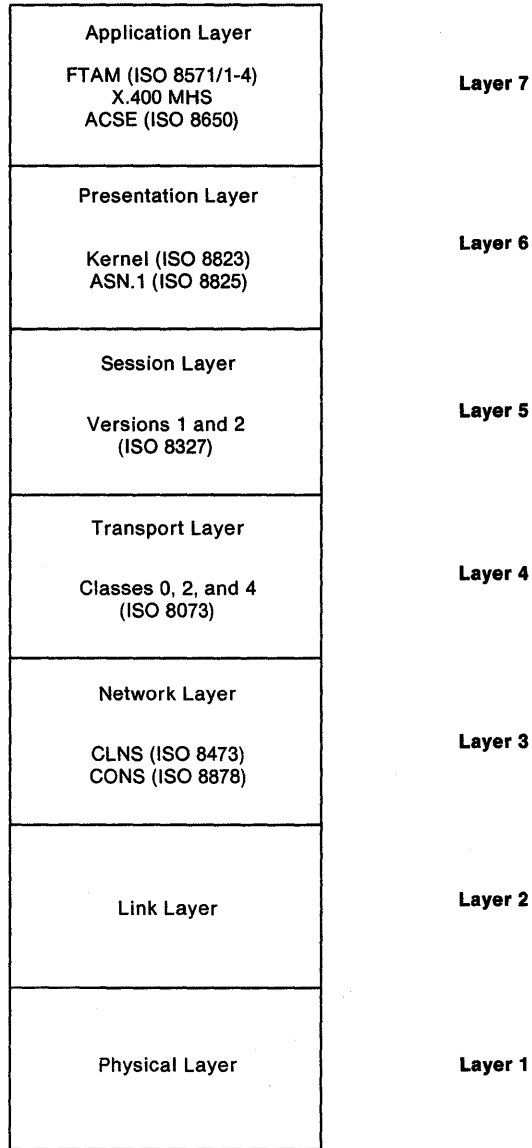
Figure 19. OSI Protocols in Common Communications Support

This remainder of the chapter discusses the OSI protocols, starting with the application layer and going through the other layers included in Common Communications Support.

Application Layer

The *Application Layer* provides access between a user application program and the remainder of the OSI environment so that the application program can exchange information with an application program in another open system.

File Transfer, Access, and Management

File Transfer, Access, and Management (FTAM) is the OSI standard (8571) for transferring and accessing files among heterogeneous computer systems. Users of similar FTAM services can exchange files with each other.

FTAM describes the file characteristics, data, and control aspects of the files being transferred between open systems. The defined file types are unstructured, flat, and hierarchical. FTAM makes the following system differences transparent to the application:

- Encoding of data

- Access method commands

- Physical organization of files.

One system initiates a file transfer between itself and another system, the *responder* system. The responder system provides access to the requested file. After a dialog has been established between the initiator and responder, the initiator can send files to, and retrieve files from, the responder.

"Related Information on International Standards" on page 65 contains a list of the international standards for FTAM. Additional information on IBM's interface to FTAM can be found in *OSI/File Services General Information*, GH19-6636.

X.400 Message Handling System

The X.400 Message Handling System is the 1984 version developed by the CCITT (The International Telegraph and Telephone Consultative Committee). This version is the initial OSI standard for interchanging electronic mail among open systems. The standard defines structured formats and protocols for interpersonal messages being transferred between end users.

A message handling system consists of a message transfer system and user agents. In the message handling system, a *user* is either a person or an application program, and is referred to as an *originator* when sending a message or a *recipient* when receiving one.

The originator prepares messages with the assistance of the originator's user agent. User agents interact with the message transfer system and provide services that allow individual users to:

- Submit messages for delivery to one or more recipients

- Receive and view messages from other users of the system.

The message transfer system comprises a number of message transfer agents (MTAs). MTAs can be compared to a postal system in the sense that their primary responsibility is to deliver messages that have been submitted for delivery to a list of intended recipients. MTAs relay and deliver messages to recipient user agents, which in turn make the messages available to the intended recipients.

Each MTA of the message handling system provides services to distinct user agents, performing such functions as:

- Accepting responsibility for delivery of messages submitted by user agents

- Relaying the messages in a store-and-forward manner to the MTAs of the recipients

- Delivering messages to recipient user agents.

Figure 20 shows the X.400 network model consisting of user agents (UAs), message transfer agents (MTAs), and the message transfer services.
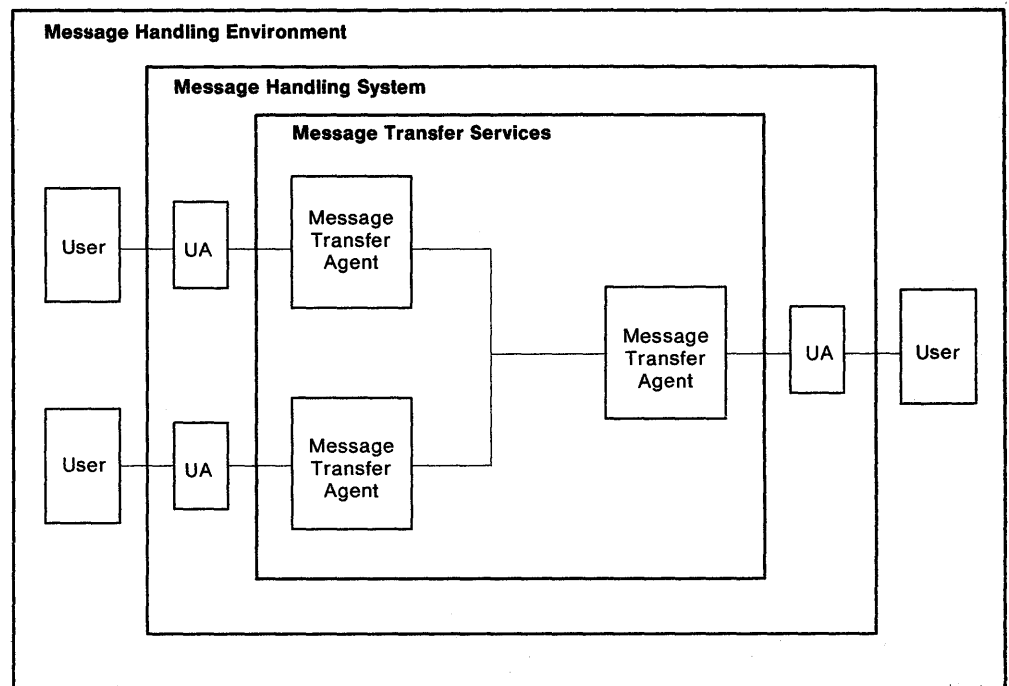
Figure 20. Model of an X.400 Message Handling System

Association Control Service Element

Association Control Service Elements (ACSEs) are those service elements considered to have a broad usefulness and which are likely to be required by most applications programs, whatever their reason for communicating. (An *association* is the logical network-wide connection between two service elements that must exist before communication can occur.) In a program, ACSEs would be analogous to common subroutines that need to be present in every program.

The ISO international standard 8650 specifies the protocol for association control. The standard describes two classes of service that are applicable for connection-oriented operations.

- *Class 1* provides the association control facility and the information transfer facility mandatory service elements.

  - The *association control facility* allows application associations to be initiated, maintained, and released.

- The *information transfer facility* allows information for two associated application entities to be transferred in standardized ways within a given context.

- *Class 2* provides the service elements defined in Class 1. In addition, it provides the context management facility service elements that have been identified as mandatory in the standard. Since class 2 includes context manipulation, it can be used where the context may need to be switched.

  - The *context manipulation facility* allows the application entity to use more than one application context over a single association.

Presentation Layer

The *Presentation Layer* selects the syntax that the invoked application program requires in order to transfer meaningful data to another application program. Either end-user system can use the data as transferred or perform data translations as required by the application program. The portions of the Presentation Layer included in Common Communications Support are:

- Kernel—establishes connection, transfers data, and terminates presentation connection. (ISO 8823)

- Abstract Syntax Notation One (ASN.1)—(1) the notation used to define a variety of data types and structures, and (2) an encoding used to transfer that data between open systems. The ISO standard for Basic Encoding Rules for ASN.t is 8825.

Additional information on IBM's interface to the Presentation Layer can be found in *OSI/Communications Subsystem General Information*, GL23-0184.

Session Layer

Because the Presentation Layer simply performs a data transformation function on behalf of the Application Layer, the *Session Layer* effectively forms the logical interface between the Application Layer and the underlying communications-oriented layers. The Session Layer allows the dialog between application programs to be organized, synchronized, and managed. Its function is to build on the basic message transport services provided by the Transport Layer to provide an end-to-end communication path between two application service processes for the duration of each complete application layer activity or transaction.

Common Communications Support includes all functional units for versions 1 and 2 of the Session Layer; additional information is in the international standard, ISO 8327. Additional information on IBM's interface to the Session Layer can be found in *OSI/Communications Subsystem General Information*, GL23-0184.

Transport Layer

The *Transport Layer* provides transparent transfer of data between open systems. It provides facilities to ensure end-to-end data integrity. The three classes of service that are part of Common Communications Support are:

- Class 0: Simple (no error recovery)

  This class provides the minimum data integrity. Only functions establishing connection, data transfer with segmenting, and error reporting are available. No exchange of user data is permitted while the connection is being established.

- Class 2: Multiplexing

  This class allows several transport connections within a single network connection. Flow control can be used as an option of class 2 to reduce congestion, optimize response times, and make best use of resources. No functions are provided for error detection or error recovery.

- Class 4: Error detection, recovery, multiplexing.

  This class provides all of the services of class 2, plus error detection and error recovery.

ISO 8073 is the standard for the Transport Layer. Additional information on IBM's interface to the Transport Layer can be found in *OSI/Communications Subsystem General Information*, GL23-0184.

Network Layer

The *Network Layer* specifies how communication should occur within and between subnetworks. It defines all possible routes that a message could take to its destination using either:

- Connectionless Network Services using Internet (ISO 8473)

  With this type of service, each information frame is a self-contained entity that is transferred using a "best-try" approach. It is commonly used with local area networks because their data-transfer integrity is high.

- Connection-oriented Network Services using Subnetwork Interface to X.25 (ISO 8878)

  This type of service makes every attempt to provide error-free information transfer. This service is useful in transferring data through a packet-switched data network, where data integrity may be unreliable. Connection-oriented Network Services use the X.25 protocols described in "X.25 in Packet-Switched Data Networks" on page 43.

Additional information on IBM's interface to the Network Layer can be found in *OSI/Communications Subsystem General Information*, GL23-0184.

Appendix A. Products that Implement CCS Protocols

This appendix contains tables showing IBM products that implement Common Communications Support architectures. Some of the products have been shipped; others have have been announced, but not shipped.

Each implementation table has headings—**TSO/E, CMS, OS/400, OS/2 EE, IMS, CICS**. When products in one of these operating environments implement an architecture, the product names are listed. When the operating environment itself implements an architecture, the operating-environment name is shown.

The abbreviations used in the tables are:

ACSE	Association Control Service Element
AS/400	Application System/400
CICS	Customer Information Control System
CMS	Conversational Monitor System (VM)
CPI-C	Common Programming Interface for Communications
DDM	Distributed Data Management
DIA	Document Interchange Architecture
DISOSS	Distributed Office Support System
DW	DisplayWrite
FOCA	Font Object Content Architecture
FTAM	File, Transfer, Access, and Management
GDDM	Graphic Data Display Manager
GOCA	Graphics Object Content Architecture
GTM OSI	General Teleprocessing Monitor for OSI
IHF	Image Handling Facility
IMS	Information Management System
IOCA	Image Object Content Architecture
IPDS	Intelligent Printer Data Stream
IVU	Image View Utility (GDDM)
LU 6.2	Logical Unit Type 6.2
MO:DCA	Mixed Object: Document Content Architecture
NCP	Network Control Program
NDM	NetView Distribution Manager
NPSI	Network Control Program Packet Switching Interface
OSI	Open Systems Interconnection
OSI Com SS	Open Systems Interconnection/Communications Subsystem
OSI/FS	Open Systems Interconnection/File Services

OSI/MS	Open Systems Interconnection/Message Services
OS/2 EE	OS/2 Extended Edition
OS/400	Operating System/400
PROFS	Professional Office System
PS/CICS	Personal Systems/Customer Information Control System
PS/TSO	Personal Systems/Time-Sharing Option
PSF	Print Services Facility
PTOCA	Presentation-Text Object Content Architecture
RFT:DCA	Revisable-Format Text: Document Content Architecture
SDLC	Synchronous Data Link Control
SNA/DS	Systems Network Architecture/Distribution Services
SNA/MS	Systems Network Architecture/Management Services
TSO/E	Time-Sharing Option/Extensions
VSE	Virtual Storage Extended
VTAM	Virtual Telecommunications Access Method
X.400 D/C	X.400 DISOSS Connection
X.400 MTF	X.400 Message Transfer Facility
X.400 P/C	X.400 PROFS Connection
3270DS	3270 Data Stream
3812-2	3812 Model 2 Page Printer
3816	3816 Page Printer
3820	3820 Page Printer
3825	3825 Page Printer
3827	3827 Page Printer
3835	3835 Page Printer
4224	4224 Printer
4234	4234 Printer
9370 ICA	9370 Integrated Channel Adapter

DDM

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
		OS/400[10]			CICS/DDM/MVS[11] CICS/DDM/VSE[11]

Note: DDM access can be through languages of the Common Programming Interface.

DIA

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
PS/TSO		OS/400			DISOSS PS/CICS

FOCA

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
PSF 3812-2 3820 3825 3827 3835	PSF 3812-2 3820 3825 3827 3835		OS/2 EE[12]	PSF 3812-2 3820 3825 3827 3835	

FTAM

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
OSI/FS	OSI/FS			OSI/FS	OSI/FS

GOCA

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
DW/370[13] GDDM	DW/370[13] GDDM	OS/400	OS/2 EE	DW/370[13] GDDM	DW/370[13] GDDM

IOCA

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
DW/370[13] GDDM IHF IVU	DW/370[13] GDDM IHF IVU PROFS	AS/400 Office	OS/2 EE	GDDM ImagePlus	DISOSS DW/370[13] GDDM ImagePlus IVU

---

[10] Source and target.

[11] Target only.

[12] Expressed in OS/2 EE format.

[13] Image and Graphics Feature.

IPDS

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
GDDM PSF 3812-2 3816 3820 3825 3827 3835 4224 4234	GDDM PSF 3812-2 3816 3820 3825 3827 3835 4224 4234	OS/400 3812-2 4224 4234		GDDM PSF 3812-2 3816 3820 3825 3827 3835 4224 4234	GDDM 3812-2 4224 4234

LU 6.2

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
VTAM	CPI-C VTAM	OS/400	OS/2 EE	IMS[14]	CICS VTAM

MO:DCA

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
ImagePlus			OS/2 EE	ImagePlus	ImagePlus

OSI

This table includes ACSE, Session layer, Presentation layer, Transport layer, and Network layer. FTAM is shown in the table under "FTAM" on page 61; X.400 is shown in the table under "X.400" on page 63.

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
OSI Com SS	OSI Com SS			OSI Com SS	OSI Com SS

PTOCA

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
DW/370 PSF	DW/370 PSF			PSF	DW/370 PSF

RFT:DCA

TSO/E	CMS	OS/400	OS/2 EE		
DW/370	DW/370	AS/400 Office	DW 4/2 DW 5/2	DW/370	DW/370

---

[14] The LU 6.1 to LU 6.2 Adapter provides IMS with a subset of LU 6.2 functions.

SDLC

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
NCP	NCP 9370 ICA	OS/400	OS/2 EE	NCP	·NCP

SNA/DS

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
NDM		OS/400			DISOSS NDM

SNA/MS

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
NetView VTAM	NetView VTAM	OS/400	OS/2 EE	NetView VTAM	NetView VTAM

Token-Ring

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
NCP VTAM	NCP VTAM w/9370 ICA	OS/400	OS/2 EE	NCP VTAM	NCP VTAM

Type 2.1 Node

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
NCP VTAM	NCP VTAM	OS/400	OS/2 EE	NCP VTAM	NCP VTAM

X.25

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
NCP/NPSI VTAM	NCP/NPSI VTAM VTAM w/9370 ICA	OS/400		NCP/NPSI VTAM	NCP/NPSI VTAM

X.400

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
X.400 MTF	X.400 MTF X.400 P/C			GTM OSI X.400 MTF	GTM OSI X.400 D/C X.400 MTF

3270DS

TSO/E	CMS	OS/400	OS/2 EE	IMS	CICS
GDDM TSO/E	GDDM CICS	OS/400[15]	OS/2 EE	GDDM IMS	CICS GDDM

---

[15] AS/400 supports the 3270 Data Stream from 3174 controllers attached via a remote line or token ring. Conversion to and from the 5250 data stream is handled internally in the OS/400. The OS/400 can pass a 3270 data stream through to a System/370 computer.

Bibliography

The IBM manuals listed in this section can be ordered from Mechanicsburg (contact your IBM representative at your local IBM branch office). They also can be obtained through an authorized IBM dealer. Additional information about ordering SAA publications is under "Additional Reading" on page 66.

Prerequisite Manuals

The following manuals contain background information on SAA, SNA, and OSI. Refer to these manuals if you are unfamiliar with basic concepts in these areas.

GC26-4341 *Systems Application Architecture: An Overview*

GC30-3072 *Systems Network Architecture: Concepts and Products*

GC30-3073 *Systems Network Architecture: Technical Overview*

ISO 7498 *Information Processing Systems—Open Systems Interconnection—Basic Reference Model*

Related Manuals for IBM Protocols

Manuals in the following list describe the IBM protocols in Common Communications Support.

SC23-0758 *Document Content Architecture: Revisable-Form-Text Reference*

GA23-0059 *3270 Information Display System Data Stream Programmer's Reference*

S544-3417 *Intelligent Printer Data Stream Reference*

SC23-0781 *Document Interchange Architecture: Technical Reference*

SC30-3098 *SNA/Distribution Services Reference*

SC30-3346 *SNA/Management Services Reference Logic for LU Type 6.2*

GC30-3084 *SNA Transaction Programmer's Reference Manual for LU Type 6.2*

SC31-6808 *SNA LU 6.2 Reference: Peer Protocols*

SC30-3422 *SNA Type 2.1 Node Reference*

SC30-3409 *The X.25-1984 Interface for Attaching SNA Nodes to Packet-Switched Data Networks - Architecture Reference*

GA27-3345 *The X.25-1984 Interface for Attaching SNA Nodes to Packet-Switched Data Networks: General Information Manual*

GA27-3761 *The X.25 Interface for Attaching SNA Nodes to Packet-Switched Data Networks: General Information Manual*

GA30-3093 *IBM Synchronous Data Link Control Concepts*

SC30-3374 *IBM Token Ring Network Architecture Reference*

SC21-9526 *Distributed Data Management Level 2 Architecture Reference*

SC21-9529 *Distributed Data Management Level 2 Implementation Programmer's Guide*

GC21-9527 *Distributed Data Management Level 2 General Information*

GA27-3136 *SNA Formats*

GG24-3296 *Architectures for Object Interchange*

Related Information on International Standards

Documents in the following list describe the international standards that are a part of Common Communications Support.

ISO 8878 *Information Processing Systems — Data Communications — Use of X.25 to Provide the OSI Connection-Oriented Network Service*

ISO 8473 *Information Processing Systems - Data Communications — Protocol for Providing the Connectionless-Mode Network Service*

ISO 8571-1 *Information Processing Systems - Open Systems Interconnection — File Transfer, Access Management - Part 1 - General Description*

ISO 8571-2 *Information Processing Systems - Open Systems Interconnection — File Transfer, Access and Management - Part 2 - The Virtual Filestore*

ISO 8571-3 *Information Processing Systems - Open Systems Interconnection — File Transfer, Access and Management - Part 3 - The File Service Definition*

ISO 8571-4 *Information Processing Systems - Open Systems Interconnection — File Transfer, Access and Management - Part 4 - The File Protocol Specification*

ISO 8650 *Information Processing Systems - Open Systems Interconnection — Protocol Specification for Association Control*

ISO 8823 *Information Processing Systems - Open Systems Interconnection — Connection-Oriented Presentation Protocol Specification*

ISO 8825 *Information Processing Systems - Open Systems Interconnection — Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)*

ISO 8073 *Information Processing Systems - Open Systems Interconnection — Connection-Oriented Transport Protocol Specification*

ISO 8327 *Information Processing Systems - Open Systems Interconnection — Basic Connection Oriented Session Protocol Specification*

ISO 8802-2 *Information Processing Systems — Local Area Networks — Part 2: Logical Link Control*

ISO 8802-5 *Information Processing Systems — Local Area Networks — Part 5: Token Ring Access Method and Physical Layer Specification*

Additional information on High-Level Data Link Control can be found in the following documents:

ISO 3309 *Data Communications—High-Level Data Link Control Procedures — Frame Structure*

ISO 4335 *Data Communications—High-Level Data Link Control Elements of Procedures*

ISO 7809 *Data Communications—High-Level Data Link Control Procedures — Consolidation of Classes of Procedures*

The following books give an overview of IBM's interface to OSI protocols.

GH19-6636 *OSI/File Services General Information*

GL23-0184 *OSI/Communications Subsystem General Information*

Additional Reading

The following books contain additional information about the Common Programming Interface and Common User Access portions of SAA. The entire set of Systems Application Architecture publications can be ordered by specifying the bill-of-forms number SBOF-1240.

SC26-4362 *Writing Applications: A Design Guide*

SC26-4582 *Common User Access: Advanced Interface Design Guide*

An additional new manual for Common User Access will be available later in 1989—*SAA Common User Access: Basic Interface Design Guide.*

Each of the following books describes an element of the Common Programming Interface.

SC26-4355 *Application Generator Reference*

SC26-4353 *C Reference*

SC26-4354 *C Reference—Level 2*

SC09-1308 *COBOL Reference*

SC26-4399 *Communications Reference*

SC26-4348 *Database Reference*

SC26-4356 *Dialog Reference*

SC26-4357 *FORTRAN Reference*

SC26-4359 *Presentation Reference*

SC26-4358 *Procedures Language Reference*

SC26-4349 *Query Reference*

SC09-1286 *RPG Reference*

Index

extended function base support (EBASE) 23

F

File Transfer, Access, and Management (FTAM)
    description 54
    product implementations 61
focal point, definition of 31
Font Object Content Architecture (FOCA)
    description 15
    minimum functions 16
    product implementations 61
fonts, use of 16
font, definition of 15
format units, in RFT:DCA 25
frame, definition of 41
FS10 function set 14, 25
FS20 function set 14, 25
FTAM
    See File Transfer, Access, and Management (FTAM)

G

GOCA
    See Graphics Object Content Architecture (GOCA)
graphic characters, definition of 15
graphic objects, structure of 15
Graphics Object Content Architecture (GOCA)
    description 15
    minimum functions 15
    product implementations 61

H

half-duplex, definition of 47
HDLC 46
High-Level Data Link Control (HDLC) 46

I

Image Object Content Architecture (IOCA)
    description 14
    minimum functions 14
    product implementations 61
implementations of CCS 59
information transfer facility, definition of 56
Intelligent Printer Data Stream (IPDS) 19
    command sets 20
        bar code 21
        device control 21
        graphics 21
        IO image 21
        loaded font 22
        overlay 22
        page segment 22
        text 21
    data tower support requirements 23
    description 19
    functional divisions 20

Intelligent Printer Data Stream (IPDS) *(continued)*
    minimum functions 22
    product implementations 62
International Standards Organization (ISO)
    description 51
    reference model 52
    standards in CCS 52
IOCA
    See Image Object Content Architecture (IOCA)
IPDS (Intelligent Printer Data Stream)
    See Intelligent Printer Data Stream (IPDS)
ISO reference model 52
italics, how they are used xii

K

Kernel, OSI Presentation Layer 56

L

link connection, definition of 41
link station, definition of 41
link, definition of 7
LLC (Logical Link Control) sublayer 43
Logical Link Control (LLC) sublayer 43
Logical Unit Type 6.2 (LU 6.2)
    how communication occurs 35
    minimum functions 36
    product implementations 62

M

MAC (Medium Access Control) sublayer 42
Medium Access Control (MAC) sublayer 42
message transfer agent (MTA) 54
message transfer system, description 54
messaging service, definition of 29
minimum functions
    DDM 32
    DIA 28
    FOCA 16
    GOCA 15
    IOCA 14
    IPDS 22
    LU 6.2 36
    MO:DCA 24
    PTOCA 14
    RFT:DCA 26
    SDLC 48
    SNA/DS 30
    SNA/MS 31
    Token-Ring 43
    Type 2.1 Node 40
    X.25 45
    3270 Data Stream 23
Mixed Object:Document Content Architecture
  (MO:DCA)
    description 24

T

target DDM
    optional functions   32
    required functions   32
target, definition of   32
text unit, in RFT:DCA   25
Token-Ring architecture
    description   41
    layers   42
    minimum functions   43
    multiple-ring connections   42
    product implementations   63
token-ring network, accessing   42
token, definition of   42
trademarks   iii
transaction program, definition of   35
Transport Layer in OSI
    classes of service   56
    description   56
    product implementations   62
two-way alternate transmission, definition of   46
two-way simultaneous transmission, definition of   46
Type 2.1 Node architecture
    description   39
    minimum functions   40
    product implementations   63

U

user agent (UA)   54
user, definition of in X.400   54

X

X.25
    description   43
    frames   45
    minimum functions   45
    packet-switched data network example   44
    product implementations   63
X.400 Message Handling System
    description   54
    model   55
    product implementations   63

Numerics

3270 Data Stream
    description   23
    minimum functions   23
    product implementations   64

Systems Application Architecture
Common Communications Support:
Summary
(GC31-6810-0)

**Reader's
Comment
Form**

Use this form to send in your comments about this book, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Your comments will be forwarded to the author for review and appropriate action.

Note: Do not use this form to request IBM publications. If you do, your order will be delayed because publications are not stocked at the address printed on the reverse side. Instead, you should direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to your IBM branch office.

If you want a reply, please complete the following information. If you do **not** want a reply, check this box: ☐

Name _____   Phone ( ) _____

Company _____   Job Title _____

Address _____

_____

Number of latest TNL applied to this publication: _____

Comments:

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM representative will be happy to forward your comments or you may mail them directly to the address in the Edition Notice on the back of the title page.)

GC31-6810-0

Reader's Comment Form

IBM ®

IBM ®

Systems Application Architecture Library

GC31-6810-0