Airline Control System Version 2

General Information Manual

Release 4.1

GH19-6738-09

Airline Control System Version 2

General Information Manual

Release 4.1

GH19-6738-09

Note!

Before using this information and the product it supports, be sure to read the general information under “Notices” on page vii.

Tenth Edition (April 2010)

This edition applies to Release 4, Modification Level 1 of Airline Control System Version 2, Program Number 5695-068, and to all subsequent releases and modifications until otherwise indicated in new editions.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address given below.

A form for readers’ comments appears at the back of this publication. If the form has been removed, address your comments to:

ALCS Development
2455 South Road
P923
Poughkeepsie NY 12601-5400
USA

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

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 product, program, or service is not intended to state or imply that only IBM's product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

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 Director of Licencing
IBM Corporation
500 Columbus Avenue
Thornwood, NY 10594
USA

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
Department 830A
522 South Road
Mail Drop P131
Poughkeepsie, NY 12601-5400
USA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at www.ibm.com/legal/copytrade.shtml

Microsoft, Windows, and Windows NT are trademarks of Microsoft Corporation in the United States, or other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

Other company, product, and service names might be trademarks or service marks of others.

About this book

This book presents general overview and planning information for Release 4.1 of Airline Control System (ALCS) Version 2, an IBM licensed program.

ALCS is one of a family of IBM programs designed to satisfy the needs of airlines and other industries with similar requirements for high-volume and high-availability transaction processing.

Who should read this book

This publication is intended for executives and data processing professionals wishing to choose a transaction processing system.

After reading this book, you should be able to:

- Evaluate ALCS Version 2 and determine its suitability to your installation
- Understand the benefits obtained by using ALCS Version 2
- Prepare for the installation of ALCS Version 2

Related publications

The ALCS Version 2 library contains the following task-oriented publications:

Application Programming Guide
: This manual describes how to design, code, execute, debug and test application programs written in assembler, COBOL, PL/I, or C language.

Application Programming Reference – Assembler Language
: This manual provides an assembler language reference which is structured alphabetically and shows the complete syntax and semantics of each macro and callable service.

Application Programming Reference – C Language
: This manual provides a C language reference which is structured alphabetically and shows the complete syntax and semantics of each ALCS C function.

Concepts and Facilities
: This manual explains the relationship of ALCS with other IBM program products. It describes the overall function of ALCS and how ALCS uses MVS and VTAM services. It explains the concept of ALCS resources and how ALCS maps its database layout on to VSAM clusters. It is also used as a link to all the other manuals in the library.

Installation and Customization
: This manual describes how to install ALCS, how to customize ALCS, how to specify the database and communication facilities for the ALCS utilities, and how to make use of installation-wide exits.

Messages and Codes
: This manual describes the messages and codes issued by ALCS. It also includes numbered error responses to the ALCS operator commands.

Operation and Maintenance

This manual describes how to operate ALCS on a day-to-day basis, how to correct system errors, and how to run ALCS utilities such as Recoup. It also provides a reference guide to the ALCS operator commands.

The "Bibliography" on page 73 gives the full title and form number for each manual and lists related publications.

The books are available in IBM Library Reader or IBM BookManager machine readable format on these IBM Online Library CD-ROMs:

- *IBM Online Library: Transaction Processing and Data*, SK2T-0730
- *IBM Online Library: OS/390 Collection*, SK2T-6700
- *IBM Online Library: z/OS Software Products Collection*, SK3T-4270
- *IBM Online Library: Messages and Codes Collection*, SK2T-2068

The CD-ROM collection kits are usually updated quarterly.

The books are also available in softcopy form on the Internet and can be displayed and printed. To find the ALCS V2 publications on the Internet, go to one of the following Web sites:

`http://www.elink.ibmlink.ibm.com/public/applications/publications/cgibin/pbi.cgi`

or

`http://www.ibm.com/software/htp/tpf/alcs/`

From the ALCS Web site above you can link to the publications by selecting **Library** from the navigation panel.

"Acronyms and abbreviations" on page 45 lists acronyms and abbreviations used throughout the ALCS library. Not every term necessarily occurs in this book.

This book also contains a glossary of terms.

Introduction

Airline Control System (ALCS) Version 2 is one of a family of IBM* programs designed to provide real-time, high-volume, and high-availability transaction processing.

The product, which is also known as TPF/MVS, provides the Transaction Processing Facility (TPF) application programming interface (API) for z/OS* environments. It supersedes ALCS/Multiple Virtual Storage/Extended Architecture (ALCS/MVS/XA*), known as ALCS Version 1.

Throughout this book, "ALCS" refers to ALCS Version 2 Release 4, unless the context makes it necessary to distinguish between ALCS Version 2 Release 4 and earlier releases.

Similarly, "z/TPF" means all versions of Transaction Processing Facility and its predecessor, Airlines Control Program (ACP).

"z/OS" refers to an IBM operating system. "MVS"* also refers to operating systems.

MQ* is now known as WebSphere* MQ, though the term MQSeries (or MQSeries for z/OS) is prevalent throughout previously published ALCS information.

Overview of ALCS

ALCS is a software interface between application programs and the z/OS operating system. It runs as a job or started task under z/OS, providing real-time transaction processing facilities for airlines, banks, hotels, and other industries that generate high transaction rates and require fast response times and high system availability.

Typical applications are passenger and cargo reservations for airlines and railroads, hotel booking systems, and credit card authorization.

ALCS provides:

- High performance and capacity
- A high level of system availability
- High transaction rates
- A wide range of communication facilities
- Connectivity with other transaction processing platforms
- Access to relational databases for business applications

Benefits include:

- z/OS environment for existing TPF applications
- z/OS data processing facilities and features
- Optimized programmer and operator productivity
- High degree of application portability between ALCS and TPF.

General description of ALCS

ALCS provides a centralized database and program system shown schematically in Figure 1. This type of system has a number of advantages over distributed systems:

- **Immediate access to data changes**

  All end users have access to current information. For example, changes in:

  – Seat availability
  – Schedules
  – Currency exchange rates
  – Account balances

  are immediately available.

- **Immediate implementation of program updates**

  Centralized program management means that all program enhancements, fixes, and other changes are immediately available to end users.

- **Central control for security**

  User access to confidential and sensitive data can be centrally controlled by facilities such as Resource Access Control Facility (RACF*). Central control also facilitates the implementation of effective disaster recovery procedures.

- **Management information**

  Current financial, business and data-processing information can be extracted for both business management and IT planning purposes. This allows for timely provision of hardware and software facilities to reflect changing business environments.

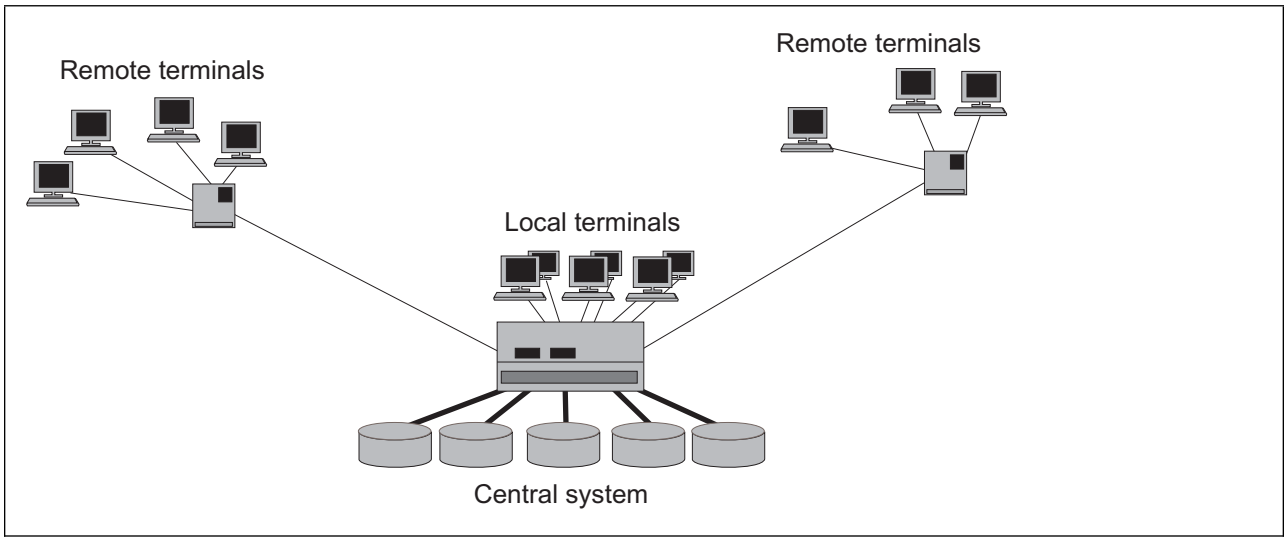

Figure 1. Centralized systems

In contrast, a typical distributed system with remote databases is shown in Figure 2 on page 3.

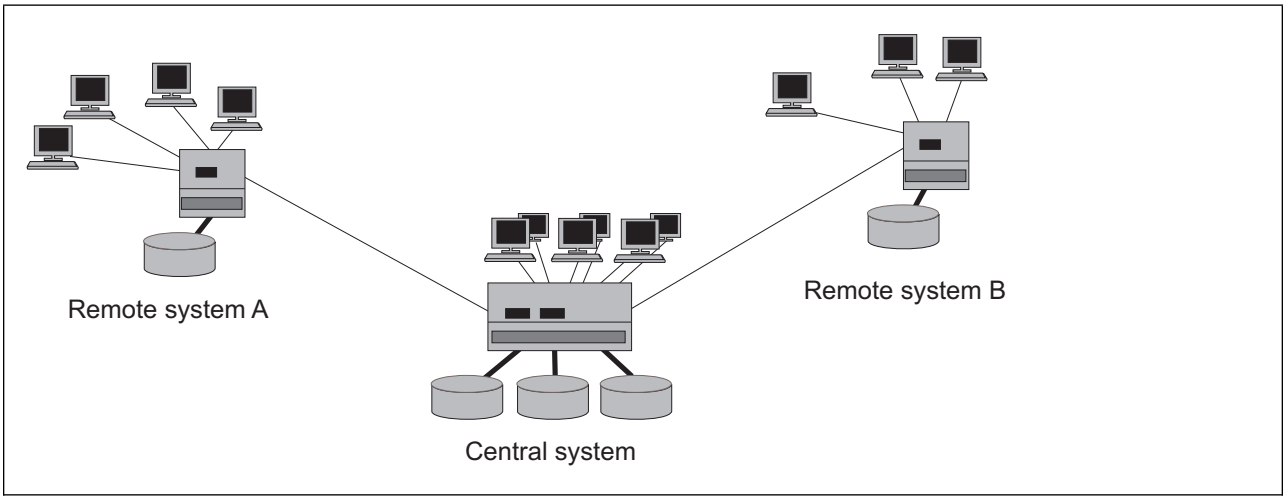

Figure 2. Distributed systems

Application programming interface

The ALCS application program interface (API) allows applications written for use with z/TPF (Version 1.1), TPF (Version 4.1 or earlier releases) and all earlier versions of ALCS to be used with few source code changes.

With ALCS, you have access to the wide range of applications developed for the TPF family of products. You can participate in an active marketplace where you can buy and sell individual applications or complete application systems.

Highlights of ALCS Version 2

This section describes the main highlights of ALCS.

Operating environment

ALCS runs under z/OS. It takes advantage of the operating system's facilities for operation, support, integrity, and security. This includes using RACF to control user access.

Storage above 2 GB, 64-bit addressing

The availability of 64-bit addressing allows ALCS to use virtual and real memory above 2 GB, removing memory constraints and freeing memory for use by applications and other subsystems.

Communication

Figure 3 on page 6 shows the range of ALCS communications.

As part of its extensive communication facilities, ALCS provides LU (Logical Unit) 6.2 communications, including Common Programming Interface – Communications (CPI-C) and Transaction Processing Facility/Advanced Program-to-Program Communications (TPPC). LU 6.2 allows the interconnection of distributed systems through APPC/MVS.

ALCS is fully compatible with WebSphere MQ for z/OS. This allows ALCS applications to exchange messages with other ALCS applications or other applications on the same system, or similar systems, or on other platforms that support message queuing using the IBM WebSphere MQ products.

ALCS can use TCP/IP to communicate with other applications in the same operating system region or with remote applications and devices.

ALCS may use the WebSphere Application Server (WAS) for z/OS optimized local adapters (OLA). These are built-in, high-speed, bi-directional adapters used for calls between WebSphere Application Sever and ALCS on the same z/OS image. This allows ALCS customers to support an efficient integration of newer Java-based applications with ALCS-based applications.

Online Communication Table Maintenance (OCTM)

This facility allows the dynamic reconfiguration of the communication network without scheduled outages.

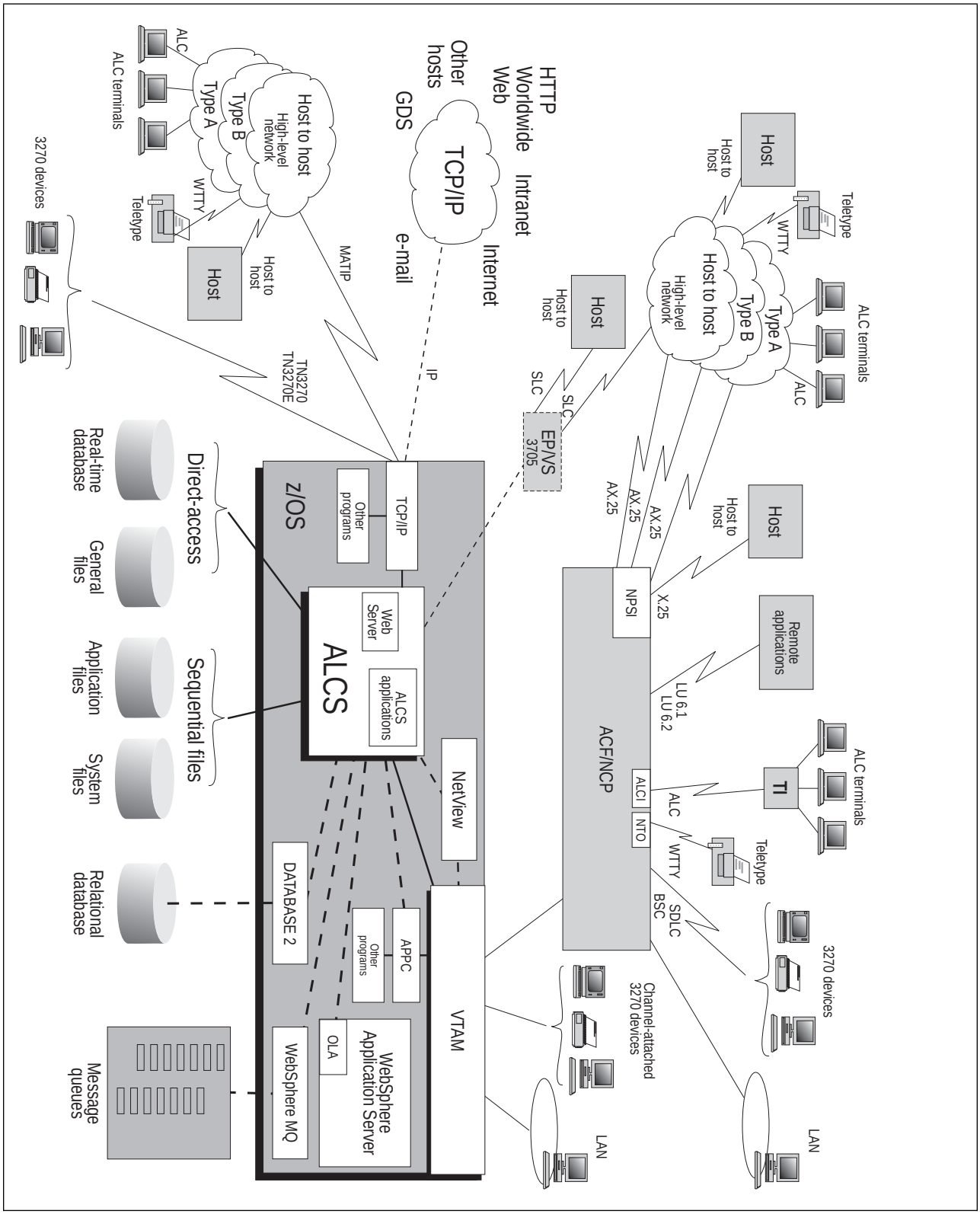

Figure 3. ALCS overview

E-mail

You can use the ALCS e-mail facility to transmit and receive e-mail messages over TCP/IP networks using an external mail server or mail transfer agent (MTA). ALCS uses the Internet Simple Message Transfer Protocol (SMTP) to communicate with the mail server.

Web Server

You can use the ALCS Web Server facility to deliver web pages and application function to Web Browser clients over TCP/IP networks. ALCS uses the Internet HTTP protocol to communicate with the clients.

Access control

Access to ALCS can be controlled by an external security manager (for example RACF). This means that users can be required to logon with a user ID and password in addition to, or instead of, the traditional AAA-based sine-in.

Automated operations

ALCS interfaces with the Tivoli NetView* family of products to allow automated operation of ALCS and its applications. This includes REXX, as supported by the installed version of NetView.

Online help

ALCS includes an enhanced context-sensitive online help facility. You can create help files specific to your location which can be integrated with the ALCS help files.

Simplified installation and operation

ALCS includes Interactive System Productivity Facility (ISPF) panels for installation, maintenance, and operating procedures. This improves usability and simplifies maintenance.

System takeover facility

The system takeover facility allows for a hot ALCS standby system, which significantly reduces the time required for planned and unplanned switchovers.

Real-time database reorganization

This feature allows online expansion, contraction, and optimization of the real-time database.

For performance reasons, accesses to an ALCS real-time database should be distributed as evenly as possible across all the direct access storage devices (DASDs) that are available to ALCS. The ALCS database generation process implements this. However, after time, an imbalance in these database accesses

can evolve. With ALCS you can redistribute database records across several DASDs without interrupting ALCS availability.

High-level language support

ALCS interfaces with IBM's Language Environment* for MVS and VM, allowing application programs to be written in the C, C++, PL/I, and COBOL languages.

For C-language programs, ALCS also provides the specialized TPF API functions, which enables portability between TPF and ALCS.

Relational database support

ALCS interfaces to the IBM DB2* for z/OS product to provide full support for Structured Query Language (SQL*), including dynamic SQL. This allows application programs read and write access to data in relational databases, including distributed databases as supported by the installed version of DB2 for z/OS.

WebSphere MQ for z/OS interface

ALCS provides full support for WebSphere MQ for z/OS. This allows ALCS applications to exchange messages with other ALCS applications or applications that are:

- On the same system
- On another platform that supports message queuing using the IBM WebSphere MQ products.

ALCS uses the common programming interface for message queueing (MQI). MQI allows programs in a heterogeneous network to exchange messages independently of network communication protocols. Applications that interact across a variety of hardware and software platforms can be developed.

This function allows ALCS to participate in an integrated transaction processing complex that can include ALCS, Customer Information and Control System (CICS*), Information Management System (IMS*), and possibly other transaction processing platforms.

Messages are passed through an initiation queue and an application queue. Figure 4 on page 9 illustrates part of this process.

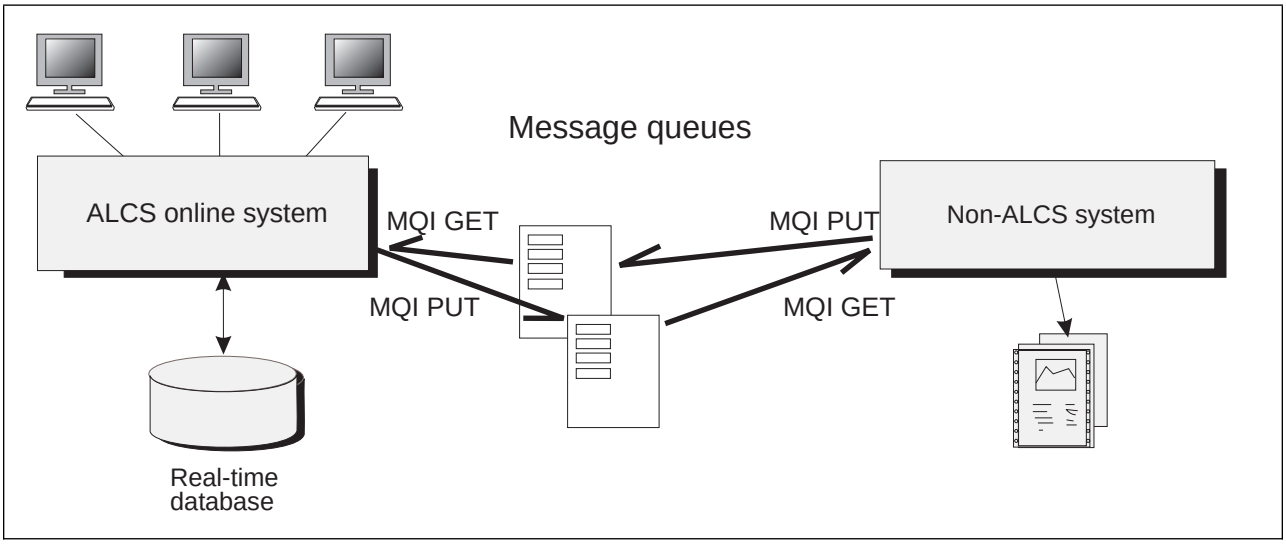

Figure 4. WebSphere MQ allows ALCS to share data

In addition, the ALCS MQ Bridge facility allows application programs to send and receive messages using MQ queues without the need to code MQI calls in ALCS programs

Optimized Local Adapters for WebSphere Application Server for z/OS

ALCS may use the WebSphere Application Server for z/OS optimized local adapters (OLA). These are built-in, high-speed, bi-directional adapters for calls between WebSphere Application Server for z/OS and ALCS in another address space on the same z/OS image that allow ALCS customers to support an efficient integration of newer Java-based applications with ALCS-based applications.

A set of callable services can be used by ALCS assembler or C/C++ programs for exchanging data with applications running in WebSphere Application Server for z/OS. For more information on the callable services (with names of the form BBOA1*xxx*) see the IBM Information Center for WebSphere Application Server - Network Deployment z/OS and search for BBOA1. The ALCS WAS Bridge installation-wide monitor exit USRWAS1 can be used to verify that the oiginator has sufficient authority.

The ALCS WAS Bridge allows ALCS application programs to send and receive messages using OLA without the need to code those callable services in ALCS programs. The ALCS WAS Bridge installation-wide monitor exits USRWAS3, USRWAS4, USRWAS5, and USRWAS6 allow you to customize the behavior of the WAS Bridge to suit your applications.

Operating environment

ALCS runs as a job or started task under z/OS. It operates in a single region, allowing other regions to be used for further ALCS systems or other applications. It uses standard operating and support facilities, and also takes advantage of the operating system's integrity and security features.

A typical ALCS operating environment is shown in Figure 3 on page 6.

ALCS can access the operating system's expanded memory and multiprocessor configurations. It is independent of operating system releases.

ALCS also benefits from many of the standard data processing facilities of its operating system. For example, its sequential files can be held on any sequential file medium supported by the MVS Data Facility Product (MVS/DFP*) including direct access storage devices (DASD).

End-user facilities

ALCS includes a number of end-user facilities that are designed to make operation easy and consistent for users.

- **Automated operations**

  ALCS interfaces with the Tivoli NetView family of products to allow automated operation of ALCS and its applications. Full REXX is supported.

  Automated operations can provide comprehensive system monitoring, message highlighting and many more functions to ensure greater system and network availability and reduce operator costs.

- **ISPF panels**

  ISPF panels provide a simple, consistent end-user interface for tasks such as installation, maintenance, and day-to-day operations. This increases usability and productivity and reduces operator training costs.

- **3270 screen mapping**

  IBM 3270 mapping support makes it easier to produce screens specific to an individual user or to an application, thus increasing user productivity.

- **Command set**

  The ALCS operator command set is small and easy to learn. Online help is also available.

- **Command confirmation**

  This facility allows users to confirm critical ALCS operator commands before they are processed. It can also be used within user applications.

- **User-definable program function (PF) keys**

  Through this facility, users can assign specific functions to certain keys. The functions may be unique to individual users or to particular applications. Function keys can simplify and speed up many user tasks.

- **Online help facility**

  The help facility provides context-sensitive help. You may add location-specific help as required.

- **Retrieve facility**

  Users can recall and edit commands they have entered. This improves operator productivity.

- **Display scrolling**

  Responses to some ALCS commands (for example display sequential file status) may be too large to fit on a single screen. ALCS provides a command that allows you to scroll to different parts of the output file containing the response. Support for large sized 3270 displays increases usability.

- **Dedicated terminals or consoles**

  Terminals may be specified as Prime Computer Room Agent Sets (Prime CRAS) or Receive Only (RO CRAS), and dedicated to ALCS operation. NetView and MVS operator IDs may also be used to operate ALCS.

- **Double byte character support**

  ALCS can work with 3270 devices that support double byte character sets. This feature is important when working with languages such as Japanese, Chinese, or Korean where each character is represented by two bytes.

Communication

ALCS supports the major communication protocols used by the airline industry as well as standard communication protocols, for example TCP/IP.

ALCS supports VTAM*, TCP/IP, and ATA/IATA SLC networks. VTAM is the IBM Virtual Telecommunications Access Method. TCP/IP is the Transmission Control Protocol / Internet Protocol. ATA/IATA SLC is the Synchronous Link Control network for the Air Transport Association of America and the International Air Transport Association.

Figure 5 provides an overview of the ALCS communication facilities.

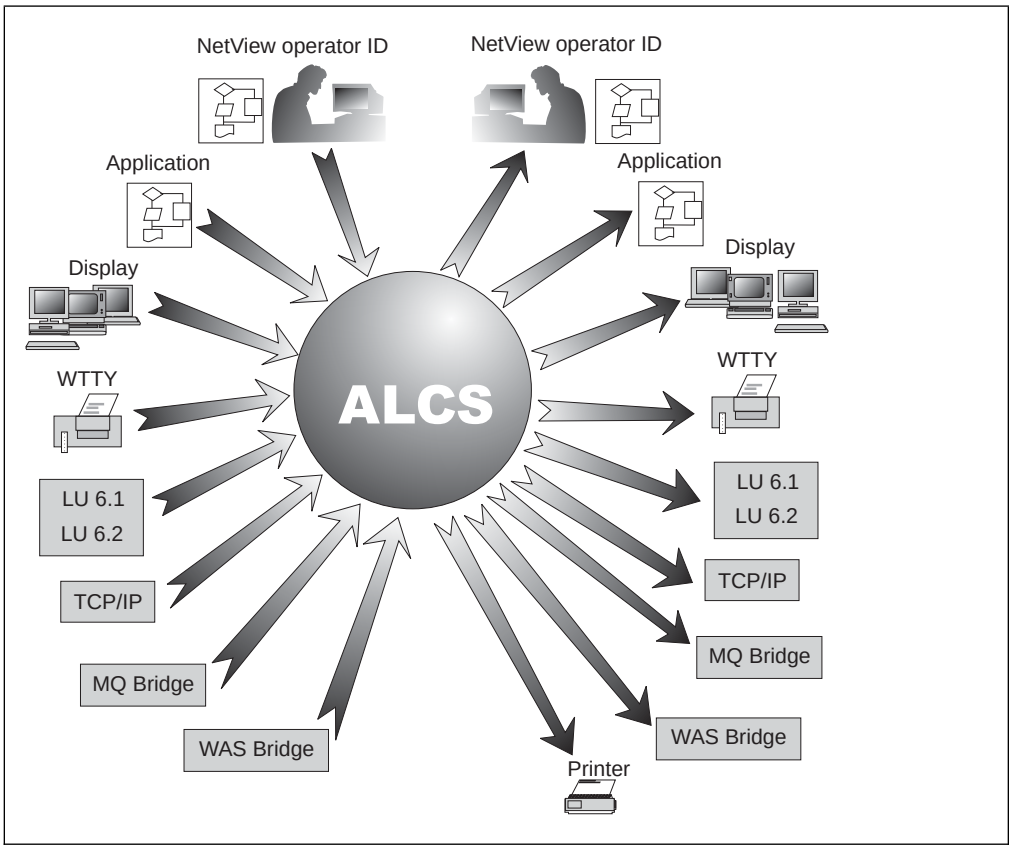

Figure 5. Communication facilities supported by ALCS

NetView can be used to provide management services for the ALCS V2 communication network.

VTAM networks

ALCS supports a wide range of VTAM networks, including token-ring LANs. It can be used with a number of different communication protocols, linking to wide area SNA* networks and local area networks.

Figure 6 shows the additional software and hardware required for each protocol.

Figure 6. VTAM networks

Network type	Protocol	Software/hardware required
SNA	X.25 (including AX.25, SITA** connection, and host-to-host)	NPSI IBM 37xx controller with ACF/NCP
SNA	LU 6.1	IBM 37xx controller with ACF/NCP
SNA	LU 6.2 (including CPI-C and TPPC)	APPC/MVS
SNA	WTTY	NTO IBM 37xx controller with ACF/NCP
SNA	ALC	ALCI or NEF2 IBM 37xx controller with ACF/NCP
SNA	SDLC	IBM 37xx controller with ACF/NCP
SNA	Token-ring LAN SDLC	Token-ring gateway IBM 37xx controller with ACF/NCP
Local		Channel-attached controller
Local	Token-ring LAN	Token-ring gateway Channel-attached controller

TCP/IP networks

ALCS supports a variety of TCP/IP configurations. Your IBM representative can supply you with information on the corresponding hardware and software requirements.

ATA/IATA SLC networks

ALCS supports communication protocols for SLC networks. Up to 255 SLC links can be configured. Each link can contain up to seven duplex channels.

Figure 7 shows the additional software and hardware required for each protocol.

Figure 7. ATA/IATA SLC network

Network type	Protocol	Software/hardware required
SLC	Processor-to-processor	IBM 3705 controller with LICRA EP/VS or NCP/VS
SLC	Host-to-host	IBM 3705 controller with LICRA EP/VS or NCP/VS Host-to-host application
SLC	SITA P.1024 and P.1124	IBM 3705 controller with LICRA EP/VS or NCP/VS

Online monitor functions

The ALCS online monitor controls the use of the resources available to the real-time system and manages the various processes that constitute that system.

Task management

The primary function of an ALCS system is to process input messages from the communication network. ALCS processes each input message as a separate user subtask. These subtasks are called *entries*. Each entry has an *entry control block* (ECB) that controls the processing of the entry.

When ALCS receives an input message from a communication terminal, the online monitor creates an ECB. It then transfers control to the application package that performs the requested function. During the processing of a message, the application programs request monitor services such as:

- Transferring control between application programs
- Obtaining and returning storage areas
- Initiating I/O operations

ALCS provides Assembler macros and C functions to invoke these services. To optimize response times, ALCS uses a subtasking system known as *non-preemptive dispatching* to control the use of system resources and balance the system load.

In a multiprocessor configuration, ALCS further optimizes response by processing multiple entries simultaneously. To ensure the integrity of shared storage areas, ALCS provides a mechanism that prevents programs from updating the same storage areas simultaneously.

ALCS implements a local program work area. This is a portion of the ECB that will be saved and restored by ALCS across application program linkages (ENTER/BACK monitor services).

Storage management

As well as the ECB itself, an entry can use additional storage blocks called working storage blocks. There are nine storage block sizes, L0 through L8.

Size symbol	Number of bytes of application data	Notes
L0	127	Recommended for TPF compatibility
L1	381	Recommended for TPF compatibility
L2	1055	Recommended for TPF compatibility
L3	4000	Minimum
L4	4095	Recommended for TPF compatibility
L5	Up to 32K	As required
L6	Up to 32K	As required
L7	Up to 32K	As required
L8	Up to 32K	As required

Application programs obtain and release working storage blocks through monitor services. Appropriate blocks are also obtained and released when applications request ALCS I/O services. The system programmer specifies the maximum storage that any one entry can use.

Application program management

ALCS provides application program linkage services. Application programs can contain multiple entry points, known as *transfer vectors*. Each program and transfer vector has a unique four-character alphanumeric name. The system programmer link-edits these programs into a number of load modules. When ALCS is started, it loads the modules into storage using a list of module names provided by the user. Modules can also be loaded online using operator commands.

ALCS provides facilities for loading and unloading programs in a test environment. Transactions are only processed by test programs associated with the originating terminal. This allows several programmers to test their new programs or fixes simultaneously on the same ALCS system.

Application programs transfer control between each other using Assembler macros or C functions. These macros specify the program or transfer vector name as a parameter. At execution time, ALCS uses this name to locate the program in storage. Assembler application programs can be up to 32K in size. There is no restriction on the size of C programs. Up to 32 levels of program nesting are permitted.

DASD management

Programs executing under the control of ALCS can access two types of data set, real-time and general, as shown in Figure 8 on page 17. In addition ALCS uses a third kind of direct access file, which contains configuration data. Application programs cannot access a configuration data set.

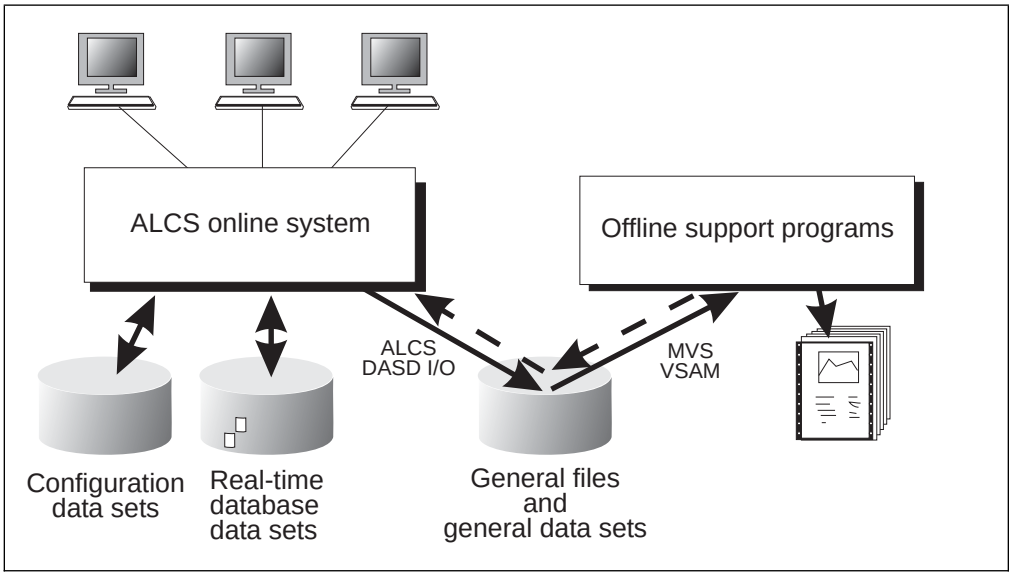

Figure 8. ALCS DASD files: Overview

ALCS supports eight DASD record sizes (L1—L8; it does not support L0 size DASD records). These eight record sizes correspond to the storage block sizes described in “Storage management” on page 15.

Real-time database data sets

These are the data sets (VSAM clusters) that the ALCS online system accesses and updates. They must be permanently available.

There are two classes of records on the real-time database:

- Fixed-file records
- Pool-file records

General files and general data sets

These are data sets that are used for communication between offline support programs and the online system. Both offline and online programs can access them, though not normally simultaneously.

The system programmer defines general files and general data sets as part of the ALCS generation process. ALCS implements them as conventional VSAM clusters. Customer-written offline programs can access these clusters using VSAM in the normal way.

Files and records

ALCS application programs use a 4-byte file address to refer to records in the real-time database data sets, general files, and general data sets. ALCS also provides support for an 8-byte file address for compatibility with TPF.

Figure 9 on page 18 provides an overview of ALCS files and records.

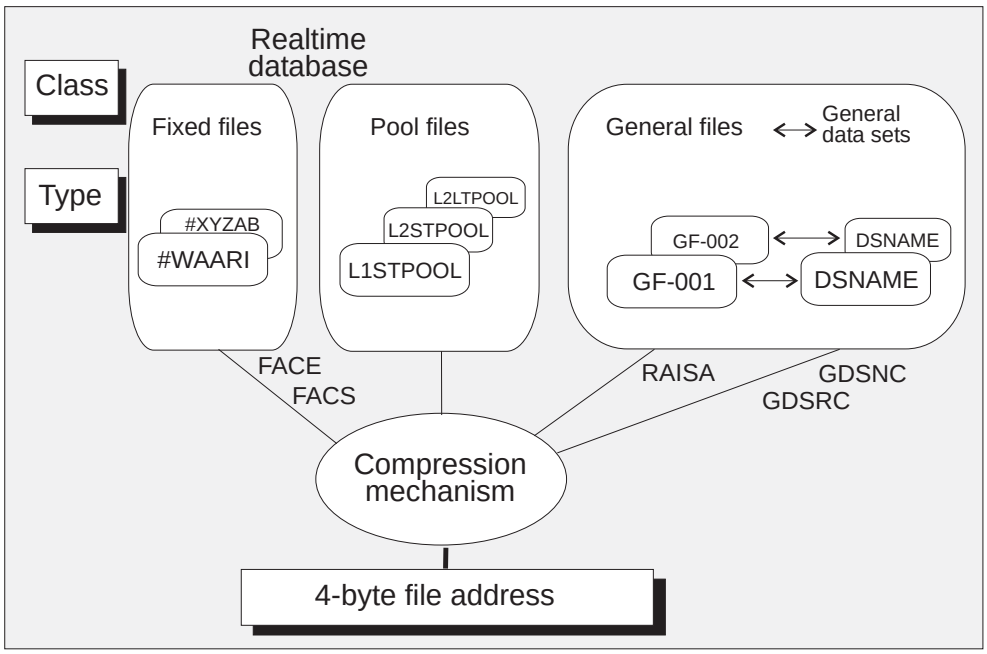

Figure 9. ALCS file address: compressing the class, type and ordinal

Fixed-file records

If it is known that the number of records of a particular type will not vary, the application programmer can use a fixed file. The system programmer is responsible for defining fixed files.

Fixed-file records on the real-time database are divided into several logical files, each containing a predetermined number of records. All the records within one logical file are the same size. The generation process specifies the number of records in each of these logical files.

The system programmer can add and delete logical files and increase and decrease the number of records within one logical file using ALCS commands. ALCS also provides facilities for recovering ("undeleting") logical files and records within logical files. The file address of a record does not depend on the physical location of that record. Typical fixed-file records are the IPARS agent assembly area (AAA) records.

Pool-file records

If the number of records may vary during execution, the application programmer can use a pool file. The system programmer is responsible for defining pool files.

There are two types of pool-file records:

Short-term For storing data that is required for a short period of time (seconds or minutes).

Long-term For storing data that is required for a long period of time (days or months).

Short and long term pools must always be defined for L1, L2 and L3 size records. Other pools need to be defined for each type and size (L4 through L8) as required by your application.

ALCS application programs access pool-file records with file addresses similar to those for fixed-file records.

To obtain a pool file record, an application program uses a monitor service that specifies a 2-byte record identifier (record ID). The record ID is used to determine the pool file record type and size. Associating the 2-byte IDs with the corresponding pool-file record sizes and types forms part of the ALCS generation procedure. When the data in a pool file record is no longer required, the application releases the record for reuse.

Typical pool-file records are the IPARS passenger name records (PNRs). The IPARS reservations application creates a PNR when a passenger makes a seat reservation. When the passenger's itinerary is complete, the PNR is no longer required and the pool-file record is released so that it can be reused.

Chains of pool-file records

An application identifies an individual pool-file record by its file address. ALCS supplies this file address when it dispenses the pool-file record to an application. Applications normally record details of each pool-file record that they use by putting the file address of the record in another record, creating a *chain* of two or more records. This address is a *pointer* to the record. See Figure 10 which shows forward and backward chains.

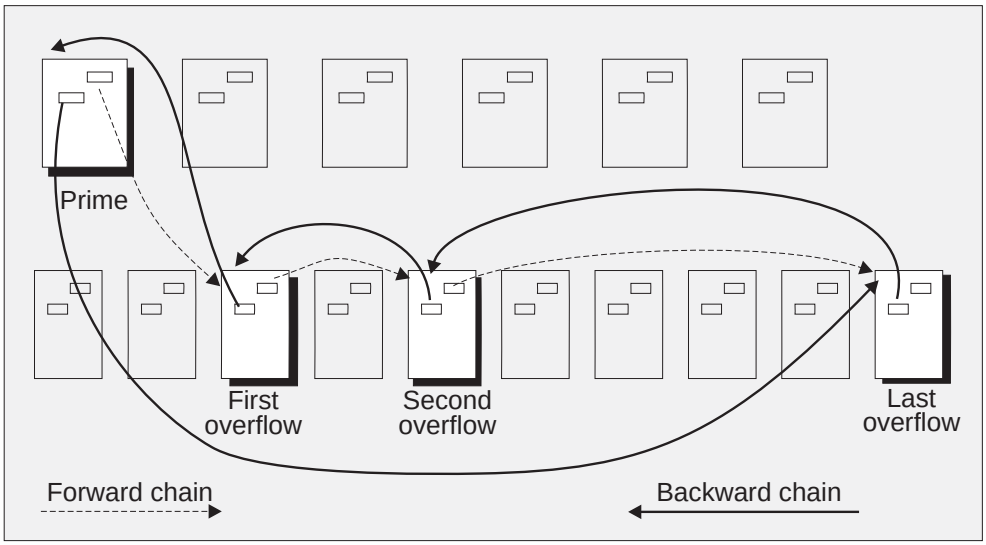

Figure 10. Chains of records

A record can contain many pointers to other records.

Records in general files and general data sets

The storage of records on general files and general data sets is analogous to the storage of fixed records on the real-time database. The concept of pool file records does not apply to general files.

Virtual file access

ALCS reduces the number of I/O operations by using *virtual file access* (VFA) to process database accesses by application programs. When using VFA, the system holds the most recently used database records in buffers (*caches*) in processor storage.

VFA combines the required level of data integrity and the minimum number of I/O operations by handling records according to attributes specified at generation time. Different types of records have different attributes, determining whether changed records are:

- Written to DASD immediately
- Deferred for a short time
- Held permanently in the buffer

For example, IPARS AAA records are referred to and perhaps updated several times by the application for every transaction from the associated terminal. Holding these records in VFA buffers provides faster response and greatly reduces the physical I/O for active terminals. The record is not written to DASD until the buffer is required by more active records.

Recoup and Pool Directory Update

A short-term pool-file record can be reused immediately after the application program releases it.

ALCS does not automatically reuse long-term pool-file records. These are only made available for reuse by Pool Directory Update (PDU) or by running Recoup. Recoup is a database validation routine that runs under the control of the ALCS online monitor, without interrupting normal message processing. Based on parameters provided by the database administrator, Recoup checks all chains of long-term pool file records and identifies the records that can safely be used.

ALCS PDU is a process that automatically reuses records when a pool is exhausted.

TPF Database Facility product

The TPF Database Facility (TPFDF) product is a database manager for TPF and ALCS applications.

ALCS application programs that use TPFDF are not sensitive to the physical implementation of the database. For example, TPFDF automatically handles overflow from fixed or pool records to pool records, retrieval of information using search keys, and so on. In addition, TPFDF can perform such functions as sorting items (logical records) by search key, and constructing and using indexes to speed up retrieval by search key.

TPFDF therefore provides:

- Increased application programmer productivity. Programmers need not develop their own algorithms for overflow, sorting, indexing, and so on.
- Improved system management capability. A database administrator can decide optimum record sizes, indexing techniques, and so on. TPFDF maintains this kind of information centrally rather than distributed through the application code.

- Easier portability to and from TPF and other ALCS systems. Applications do not need to be changed to use different record sizes.

Sequential file management

Sequential files in ALCS can be held on any sequential file media supported by MVS/DFP. These include magnetic tape, DASD, SYSOUT data sets, partitioned data set (PDS) members, and generation data groups. There are two types of sequential files:

- System sequential files
- Application sequential files

Figure 11 provides an overview of sequential file management in ALCS.

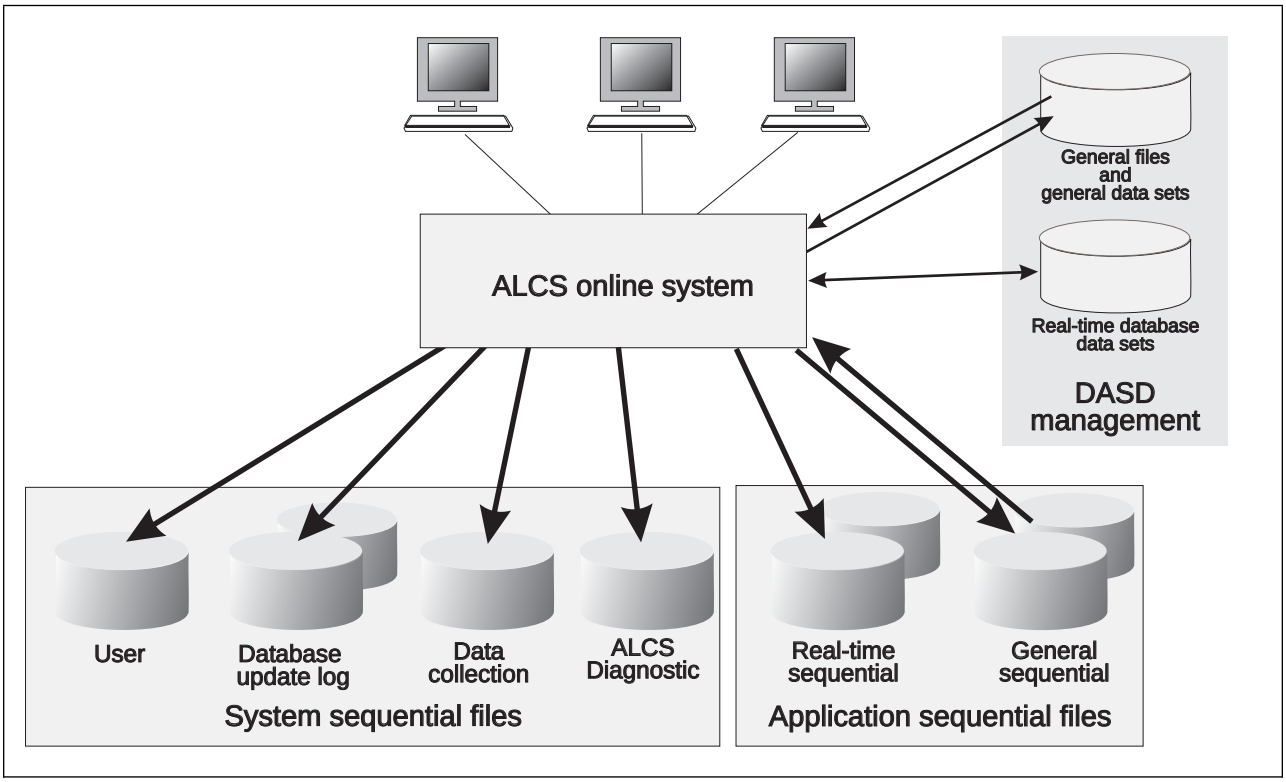

Figure 11. Sequential file management in ALCS

System sequential files

These files are for the exclusive use of the ALCS monitor:

- **Diagnostic file**

  ALCS writes diagnostic data, such as dumps, trace information, and pool record usage errors, to the ALCS diagnostic file for subsequent offline processing. There is only one diagnostic file, which is permanently allocated.

- **Database update log**

  ALCS records changes to the real-time database in the database update log. By default this log is a single file, with an option to have two log files, a *forward log* (containing the after image) and a *backward log* (containing the before image). All database update log files are permanently allocated.

- **Data collection file**

  Statistics can be collected on various types of performance data such as I/O operations and messages, ECBs, task waits, application program use, and CPU utilization. The data is written to a system sequential file for subsequent offline processing by the ALCS statistics report generator (SRG) or similar utility. If no data collection file is defined, output is written to the ALCS diagnostic file.

- **User file**

  ALCS itself does not write to this system sequential file. It is provided for writing data from an installation-wide monitor exit, using an ALCS service. The data is written to a system sequential file for subsequent offline processing by a user-written offline program. If no user file is defined, then the records are written to the ALCS diagnostic file.

Application sequential files

These files are used only by application programs:

- **Real-time sequential files**

  A real-time sequential file (corresponding to a TPF real-time tape) is an output-only file. Application programs use ALCS monitor services to write records to the real-time sequential files. All real-time sequential files are permanently allocated.

- **General sequential files**

  A general sequential file (corresponding to a TPF general tape) can be an input or an output file. Only one entry at a time can use a general sequential file. General sequential files are allocated and deallocated at the request of the application. Before using a general sequential file an application program uses an ALCS monitor service to gain control of the file. The program has exclusive use of the file until it uses a second ALCS monitor service to release control.

Communication management

Much of the communication management is best performed by VTAM and NetView. ALCS, however, provides commands that give users additional control over their communication resources. For example:

- Displaying the status of ALCS resources
- Starting and stopping SLC links
- Starting and stopping ALCS applications
- Initiating and terminating SNA sessions

Printer support

ALCS provides a number of facilities for managing printers:

Printer shadowing: This facility allows up to 16 printers to receive a copy of a message sent to a particular printer.

Printer sharing: This facility allows 3270-type printers to be shared between ALCS and other applications.

Printer redirection: This facility allows output messages for a particular printer to be sent to another printer. Up to 16 printers can be redirected one to another in a chain.

Messages on WebSphere MQ for z/OS

ALCS provides full support for WebSphere MQ for z/OS MQI calls. The messaging, queueing, and interoperability features of MQI provide many benefits to the transaction processing environment:

- Additional connectivity between ALCS systems in different regions or under different z/OS systems
- Connectivity to other applications, for example Time Sharing Option (TSO), IMS, and CICS
- Multi-platform support

Figure 3 on page 6 shows ALCS connected to WebSphere MQ for z/OS which is a part of the IBM WebSphere MQ products.

ALCS allows applications to issue MQI calls which it intercepts and passes on to WebSphere MQ for z/OS. Similarly, the results of MQI calls are returned to the calling application. MQI calls are simple for application programmers to understand and use. Figure 4 on page 9 shows an overview of message queueing.

Access to relational databases

ALCS supports full-function SQL. This allows ALCS applications to access data held in local and remote relational databases (RDBs), as well as in the ALCS online database. See Figure 12.

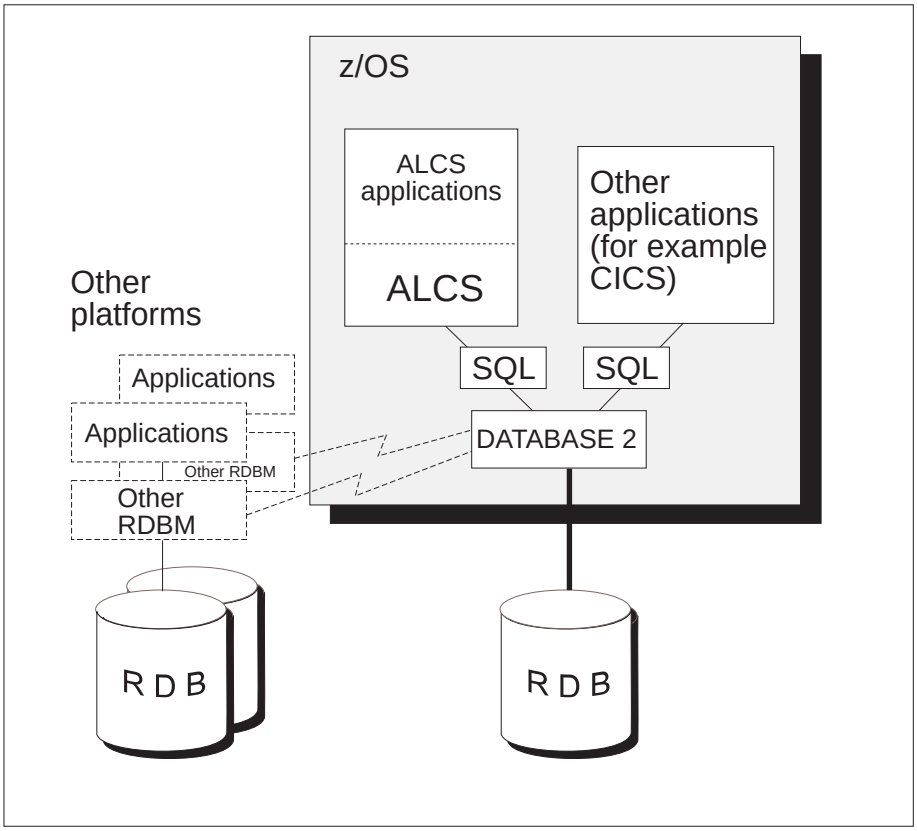

Figure 12. Relational database access

Operational data can be readily transferred to and from decision support applications, for example yield management and revenue accounting. Distributed relational databases are also accessible through SQL.

High-level languages

ALCS provides full support for IBM's Language Environment for MVS and VM products, including the C, C++, PL/I, and COBOL languages. ALCS provides C-language calls for most of the ALCS monitor services.

The ALCS Application Program Interface (API) ensures that ALCS is compatible with programs written for TPF or for earlier versions of ALCS.

All C-language programs written for TPF must be recompiled with the ALCS C header files before being run on ALCS.

Maintenance

Within ALCS, there are many facilities to ensure easy system maintenance. These include performance monitoring, testing, and recovery features. In addition, users of ALCS can access the standard maintenance features of VTAM, DFP, and its operating system.

Performance monitoring

ALCS provides the following facilities for monitoring system performance:

- **Performance monitor**

  This online function collects performance data and stores it on the ALCS real-time database. It can produce online performance reports based on current data and historical data.

- **Data collection**

  This online function collects performance data and writes it to the ALCS data collection file (or to the diagnostic file if no data collection file is defined). You can use Service Level Reporter or a similar product to process the data.

- **Online displays**

  ALCS provides several operator commands for displaying performance information online.

- **MVS workload manager**

  The MVS workload manager collects performance information from its clients and can produce performance reports for end users.

Reconfiguration

ALCS can be reconfigured online. You can use its standard operator commands to make temporary changes to configuration data. In most cases, you can make a permanent change by simply rebuilding the appropriate configuration-dependent tables.

Database organization

The ALCS real-time database can be expanded or contracted online without interrupting availability.

Options include:

- Changing the size or number of data sets in the database
- Changing the number of records in long-term pool
- Allocating fixed and short-term pool file records dynamically
- Adding or deleting records
- Restriping the database (that is redistributing the databases over several datasets) to extract maximum performance from the attached DASD devices

Communication network reconfiguration

ALCS includes an online communication table maintenance facility.

Testing facilities

ALCS provides its own facilities for maintenance testing. In addition, it has access to the testing facilities contained within z/OS and VTAM. Both sets of facilities are detailed in “Testing” on page 33.

Recovery

ALCS provides the following facilities for recovery:

- **Database duplication**

  You can configure a duplicate real-time database during ALCS generation or by using the IBM 3990 Dual Copy Facility. Both copies of the database are then automatically updated whenever a record is written. The dual copy facility can also be used to duplicate other critical data (for example, the update log).

- **Database update log**

  The ALCS monitor logs changes to the real-time database. The log can be used in conjunction with backups to recover from data loss or corruption of the database.

- **Standby ALCS**

  You can maintain a standby ALCS in case of abnormal termination. If the prime ALCS fails, the standby can assume control after a simple automated or manual command. This helps to achieve high system availability.

  In addition, ALCS includes a system takeover facility. When a standby ALCS is enabled for system takeover, it continually checks the status of the prime ALCS. When the prime ALCS terminates, the standby ALCS assumes control immediately.

- **Automatic load management**

  The processing load is monitored continuously. This ensures an automatic recovery from any transient overload.

- **Standard error recovery facilities**

  In addition to its own error recovery facilities, ALCS also has access to those provided by z/OS , VTAM, and DFP.

Testing

ALCS provides a number of its own testing facilities. In addition, it has access to the testing facilities contained within the operating system. Both sets are suitable for maintenance and development testing.

ALCS testing facilities

- **ALCS trace facility**

  This is a problem determination aid which monitors the execution of monitor services, interrupting processing at specified places to log selected data. Trace information can be written to a terminal, the diagnostic file, or the system macro trace block.

  An instruction trace is also available. The information from this kind of trace is always written to a terminal, which means that the instruction trace facility runs interactively.

  The ALCS trace facility can initiate source level debugging of C/C++ programs on a remote workstation.

- **System test vehicle**

  The system test vehicle (STV) simulates terminal input, and records input and output messages for verification. It is supported by a system test compiler. Together, they are used to test new application programs and modifications to existing programs.

- **SLC link trace facility**

  This monitors synchronous link control (SLC) link activity. It traces specified SLC link control and data blocks. Trace information can be sent to a printer, to the diagnostic file, or to the system SLC trace block.

- **SLC link test facility**

  This sets conditions for testing the SLC network. It allows you to send messages and to simulate error conditions on an SLC link.

- **TCP/IP trace facility**

  This monitors activity on TCP/IP connections. It traces inbound and outbound TCP/IP data. Trace information is written to the system TCP/IP trace block; it can also be written to the diagnostic file or a Web browser.

- **ALCS diagnostic file processor**

  This is commonly called the *post processor*. It is a problem determination aid that prints selected records from the diagnostic file. It prints formatted dumps, trace and test output, and diagnostic information about pool file usage errors.

- **Test database facility**

  This facility allows multiple ALCS test systems to share a common test database. The test database is typically a copy of part or all of the production database. Separate updates are kept so that each user's test data is protected against corruption by another user.

- **Dynamic program load facility**

This is a function for loading and unloading application programs during normal execution. It helps to prevent disruption of transactions by allowing currently active transactions to complete using the old version of the application programs. Individual users can use this facility to load their own private copies of programs during testing.

- **Recoup**

  Recoup is a real-time database validation routine that helps to identify the cause of database problems. It checks the chains of all long-term pool file records, and updates the pool file directories. See “Recoup and Pool Directory Update” on page 20 for more details.

Other testing facilities

Since ALCS exploits the z/OS environment, ALCS users can use the testing features of standard products such as VTAM and DB2. These testing facilities include:

- Dumps
- Traces
- SLIP traps
- Teleprocessing Network Simulator (TPNS)
- Generalized Trace Facility (GTF)

Integrity

ALCS provides a number of its own integrity features, which operate as part of the normal running of the system. In addition, it is able to use the integrity facilities provided within z/OS and the standard products that it uses.

ALCS features

ALCS provides the following integrity features:

- **Protected storage**

  ALCS provides a protected storage area for application programs. This prevents corruption by other programs.

  Application programs working with ALCS are issued with a program status word (PSW) key that gives them access to certain storage areas. They cannot access the storage areas of ALCS itself, and ALCS will not modify its own storage areas for them. ALCS does not provide a service to change the PSW key of any application program.

- **Recoup**

  Recoup is a real-time database verification routine. It runs without operator intervention, in any system state. It does not interrupt normal message processing.

- **Short-term pool integrity**

  ALCS includes a number of integrity features that reduce or eliminate problems such as double-release, failure to release, and records being used for too long.

- **Long-term pool integrity**

  ALCS includes a number of integrity features that reduce or eliminate the data loss caused by long-term pool corruption (for example, broken chains).

- **Process security**

  Each ALCS transaction is stored in an isolated area. This helps to prevent one transaction from corrupting another.

Security

ALCS provides its own security features, which operate as part of the normal running of the system. In addition, it is able to use the security facilities provided within z/OS (for example, Resource Access Control Facility (RACF) and the standard products that it uses.

ALCS features

Security features provided by ALCS include the following:

- **End-user access control**

  Access to ALCS can be controlled by a user ID and password (instead of, or as well as, the current AAA-based sine-in and similar access control systems). ALCS actually uses an external security manager (for example RACF) to check access authority.

- **CRAS**

  CRAS (Computer Room Agent Set) terminals can be assigned different levels of authority. Commands are routed through ALCS, which checks that the terminal issuing the command has authorization to do so. This ensures that only designated users have access to commands which could affect the security or integrity of the system.

- **DB2 access checks**

  ALCS ensures that users do not access data through DB2 for z/OS without adequate authorization. Access can be restricted to particular users or particular terminals.

- **MQ access checks**

  ALCS ensures that users do not access data through WebSphere MQ for z/OS without adequate authorization. Access can be restricted to particular users or particular terminals.

- **APPC access checks**

  ALCS ensures that users do not access data through APPC without adequate authorization. Access can be restricted to particular users or particular terminals.

- **WAS access checks**

  ALCS ensures that users do not access data through WebSphere Application Server (WAS) for z/OS without adequate authorization. Access can be restricted to particular users or particular terminals.

Installation and migration

This section describes how to install or migrate to ALCS Version 2 Release 4.

Migrating from predecessor ALCS versions and releases

Migrating from ALCS Version 2 Release 1.3, ALCS Version 2 Release 2.1, ALCS Version 2 Release 4.1, to ALCS Version 2 Release 4 is a straightforward process provided that the existing ALCS system:

- Uses only published interfaces
- Is not modified internally
- Is at the current level of maintenance

You can run the same application programs without reassembly. You can use the same database without reorganization.

You will need to reassemble installation-wide exit routines; but you may also want to make minor changes to them or add new exit routines to exploit the new functionality available in ALCS Version 2 Release 4.1.

If you are migrating from ALCS Version 2 Release 1.3 and you use the application global area, you must reassemble and relink-edit the global load control programs GOA0 through GOAE.

You can use an ALCS Version 2 Release 4 system as a standby for an older ALCS Version 2 system, which makes it easier to plan and control a staged migration as follows:

1. Install ALCS Version 2 Release 4
2. Test and configure ALCS Version 2 Release 4 and use it as a fallback system
3. Switch to ALCS Version 2 Release 4 and use the old system as a fallback

Migrating from TPF

If you are transferring from TPF, you will need to migrate the database, the application programs, and the communication facilities. Figure 13 shows what you need to do to migrate the different parts of your system.

Figure 13. Migrating to ALCS Version 2 Release 4

Database	• Use TPF/DBR to write TPF data to tape • Use ALCS data load facility to restore the database to ALCS. • ALCS supports a basic TPF file addressing format. If your format is complicated, you will need to write an ALCS exit routine to interpret the file addresses.
Applications	• Source-level compatible in most cases • All Assembler language programs must be reassembled with ALCS macrodefinitions. • C language programs must be recompiled with ALCS C header files.
Communication	• All protocols supported except: – BSC point-to-point – BSC multipoint – AT&T** 83B3 asynchronous link control – ARINC** asynchronous link control • Use existing hardware to migrate SLC networks

Installation and migration summary

If you are transferring from TPF, or if you are starting ALCS Version 2 Release 4 from scratch, the installation process is more involved. The size and complexity of the process will vary greatly from one installation to another.

Figure 14 shows the main stages involved in installing ALCS Version 2 Release 4 in each case.

Figure 14. Installing ALCS Version 2 Release 4

From ALCS 2.2.1, ALCS 2.1.3, or ALCS 2.3.1	From TPF	From scratch
• Use SMP/E to unpack tape to library • Build generation input • Run generation • Evaluate and reapply installation-wide exits • Run tests to validate application • Run tests to validate system • Bring up new ALCS to standby state • Halt the old ALCS system and switch over to the new one • Activate database reorganization and pool support (unless you have already done this in ALCS Version 2.2.1 , 2.1.3 or 2.3.1)	• Use SMP/E to unpack tape to library • Build generation input • Run generation • Create or reorganize database • Reassemble or recompile existing application programs • Develop and install installation-wide exits • Run tests to validate application • Run tests to validate system • Bring up new ALCS to standby state • Halt the old TPF system and switch over to the new one	• Use SMP/E to unpack tape to library • Build generation input • Run generation • Design and implement database • Develop and install application programs • Develop and install installation-wide exits • Run tests to validate application • Run tests to validate system • Bring up new ALCS to standby state • Activate database reorganization and pool support

Required hardware

ALCS runs on any hardware supported by z/OS provided that the processor can be configured with sufficient primary and auxilliary storage for:

- ALCS
- The z/OS operating system
- All required software
- The recommended software
- Your application programs
- Your application's in-memory tables

A coupling facility is required for Pool Directory Update.

The amount of DASD storage required depends on the amount of data the application programs use and the system's transaction rate.

Required software

This section specifies the minimum software requirements for installing ALCS without any of the optional product features. It also details the requirements for each product feature and lists optional programs that are recommended, for example, for optimizing productivity and enhancing usability.

Control program

The minimum required level of operating system for ALCS Version 2 Release 4 is z/OS (5694-A01) at all release levels.

Required software for special functions

Specific functions in ALCS require compatible versions and releases of the following products:

- WebSphere MQ for z/OS V6 (5655-L82), or later, if application programs use the message queuing interface (MQI)
- IBM WebSphere Application Server for z/OS V7.0.0.4 (5724-J08) or later, if application programs use the WebSphere optimized local adapter (OLA) calls or the WAS Bridge.
- DB2 for z/OS V8 (5625-DB2) or later, if application programs use structured query language (SQL)
- Tivoli NewView for z/OS (5697-ENV), if the installation uses Netview operator IDs to communicate with ALCS.

C applications may be compiled using:

- z/OS XL C/C++ (5694-A01)

COBOL applications may be compiled using:

- IBM Enterprise COBOL for z/OS V4.1 (5655-S71)

Recommended software

The following software products are recommended to optimize programmer productivity and the performance and security of your ALCS installation.

- **TPF Database Facility**

  TPFDF is a database manager for ALCS application programs. It increases programmer productivity by automating traditional database management functions. See “TPF Database Facility product” on page 20 for more details.

- **NetView Performance Monitor**

  NetView gives comprehensive information on the utilization of network resources. It also provides facilities for automating operations such as backing up.

- **Resource Management Facility**

  RMF* is a feature of z/OS. It provides comprehensive information on the utilization of operating system resources.

- SecureWay Security Sever Resource Access Control Facility (RACF)

  RACF is a security feature of z/OS. It protects data and programs from unauthorized access.

- **Service Level Reporter**

  SLR is an interactive tool for analyzing performance data.

- **MVS/Data Interfile Transfer, Testing and Operation Utility**

With appropriate RACF authority, you can use DITTO to browse or update VSAM clusters, including the ALCS real-time database and general file data sets. You can also use it to browse ALCS sequential files.

- **MVS/Interactive Problem Control System**

  IPCS is a diagnostic tool that formats dumps and traces online, allowing interactive problem determination and solution.

- **Advanced Application Communication System**

  A2CS provides a communication facility that allows PC workstations on a LAN to link to one or more host systems by ALC, SNA, or TCP/IP.

- **Data Facility Data Set Management Subsystem (DFSMSdss*)**
- **Data Facility Hierarchical Management (DFSMShsm*)**

Acronyms and abbreviations

The following acronyms and abbreviations are used in books of the ALCS Version 2 library. Not all are necessarily present in this book.

AAA	agent assembly area
ACB	VTAM access method control block
ACF	Advanced Communications Function
ACF/NCP	Advanced Communications Function for the Network Control Program, usually referred to simply as “NCP”
ACF/VTAM*	Advanced Communications Function for the Virtual Telecommunication Access Method, usually referred to simply as “VTAM”
ACK	positive acknowledgment (SLC LCB)
ACP	Airline Control Program
AID	IBM 3270 attention identifier
AIX	add item index
ALC	airlines line control
ALCI	Airlines Line Control Interconnection
ALCS/MVS/XA	Airline Control System/MVS/XA
ALCS/VSE	Airline Control System/Virtual Storage Extended
ALCS V2	Airline Control System Version 2
AML	acknowledge message label (SLC LCB)
AMS	access method services
AMSG	AMSG application message format
APAR	authorized program analysis report
APF	authorized program facility
API	application program interface
APPC	advanced program-to-program communications
ARINC**	Aeronautical Radio Incorporated
ASCU	agent set control unit (SITA), a synonym for “terminal control unit”
AT&T**	American Telephone and Telegraph Co.
ATA	Air Transport Association of America
ATSN	acknowledge transmission sequence number (SLC)
BATAP	Type B application-to-application program
BSC	binary synchronous communication
C	C programming language
CAF	DB2 Call Attach Facility
CCW	channel command word
CDPI	clearly differentiated programming interface
CEC	central electronic complex
CEUS	communication end-user system
CI	VSAM control interval
CICS*	Customer Information Control System
CLIST	command list
CMC	communication management configuration
CML	clear message label (synonym for AML)
COBOL	COmmon Business Oriented Language
CPI-C	Common Programming Interface – Communications
CPU	central processing unit
CRAS	computer room agent set
CRI	communication resource identifier

CRN	communication resource name
CSA	common service area
CSECT	control section
CSID	cross system identifier
CSW	channel status word
CTKB	Keypoint record B
CTL	control system error
CUA*	Common User Access
DASD	direct access storage device
DBCS	double-byte character set
DBRM	DB2 database request module
DB2*	IBM DB2 for z/OS
DCB	data set control block
DECB	ALCS data event control block
DF	delayed file record
DFDSS	Data Facility Data Set Services
DFHSM	Data Facility Hierarchical Storage Manager
DFP	Data Facility Product
DFSMS*	Data Facility Storage Management Subsystem
DFT	distributed function terminal
DIX	delete item index
DRIL	data record information library
DSI	direct subsystem interface
DSECT	dummy control section
DTP	ALCS diagnostic file processor
EBCDIC	extended binary-coded decimal interchange code
ECB	ALCS entry control block
EIB	error index byte
EID	event identifier
EJB	Enterprise Java Bean
ENQ	enquiry (SLC LCB)
EOF	end of file
EOM	end of message
EOI	end of message incomplete
EOP	end of message pushbutton
EOU	end of message unsolicited
EP	Emulation Program
EP/VS	Emulation Program/VS
ETX	end of text
EVCB	MVS event control block
EXCP	Execute Channel Program
FACE	file address compute
FIFO	first-in-first-out
FI	file immediate record
FM	function management
FMH	function management header
GB	gigabyte (1 073 741 824 bytes)
GDS	general data set
GFS	get file storage (called pool file storage in ALCS)
GMT	Greenwich Mean Time
GTF	generalized trace facility (MVS)
GUPI	general-use programming interface
HEN	high-level network entry address
HEX	high-level network exit address

HFS	Hierarchical File System
HLASM	High Level Assembler
HLL	high-level language
HLN	high-level network
HLS	high-level system (for example, SITA)
IA	interchange address
IASC	International Air Transport Solution Centre
IATA	International Air Transport Association
IATA5	ATA/IATA transmission code 5
IATA7	ATA/IATA transmission code 7
ICF	integrated catalog facility
ID	identifier
ILB	idle (SLC LCB)
IMA	BATAP acknowledgement
IMS*	Information Management System
IMSG	IMSG input message format
I/O	input/output
IOCB	I/O control block
IP	Internet Protocol
IPARS	International Programmed Airlines Reservation System
IPCS	Interactive Problem Control System
IPL	initial program load
ISA	initial storage allocation
ISC	intersystem communication
ISO/ANSI	International Standards Organization/American National Standards Institute
ISPF	Interactive System Productivity Facility
ISPF/PDF	Interactive System Productivity Facility/Program Development Facility
ITA2	International Telegraph Alphabet number 2
JCL	job control language
JES	job entry subsystem
JNDI	Java Naming and Directory Interface
KB	kilobyte (1024 bytes)
KCN	link channel number (SLC)
KSDS	VSAM key-sequenced data set
LAN	local area network
LCB	link control block (SLC)
LDB	link data block (SLC)
LDI	local DXCREI index
LEID	logical end-point identifier
LE	Language Environment*
LICRA	Link Control – Airline
LMT	long message transmitter
LN	line number (ALCS/VSE and TPF terminology)
LN/ARID	line number and adjusted resource identifier (ALCS/VSE terminology)
LSI	link status identifier (SLC)
LU	logical unit
LU 6.2	Logical Unit 6.2
MATIP	Mapping of airline traffic over IP
MB	megabyte (1 048 576 bytes)
MBI	message block indicator (SLC)
MCHR	module/cylinder/head/record

MESW	message switching
MNOTE	message note
MQI	Message Queueing Interface
MQM	Message Queue Manager
MSNF	Multisystem Networking Facility
MVS*	Multiple Virtual Storage (refers to both MVS/XA and MVS/ESA, and also to OS/390* and z/OS*)
MVS/DFP*	Multiple Virtual Storage/Data Facility Product
MVS/ESA*	Multiple Virtual Storage/Enterprise System Architecture
MVS/XA*	Multiple Virtual Storage/Extended Architecture
NAB	next available byte
NAK	negative acknowledgment (SLC LCB)
NCB	network control block (SLC)
NCP	Network Control Program (refers to ACF/NCP)
NCP/VS	Network Control Program/Virtual Storage.
NEF	Network Extension Facility
NEF2	Network Extension Facility 2
NPDA	Network Problem Determination Application
NPSI	Network Control Program packet switching interface
NTO	Network Terminal Option
OCR	one component report
OCTM	online communication table maintenance
OLA	optimized local adapters
OMSG	OMSG output message format
OPR	operational system error
OSID	other-system identification
OS/2*	IBM Operating System/2
PARS	Programmed Airlines Reservation System
PDF	parallel data field (refers to NCP)
PDM	possible duplicate message
PDS	partitioned data set
PDSE	partitioned data set extended
PDU	pool directory update
PER	program event recording
PFDR	pool file directory record
PL/I	programming language one
PLM	purge long message (name of ALCS/VSE and TPF general tape)
PLU	primary logical unit
PNL	passenger name list
PNR	passenger name record
PP	IBM program product
PPI	program-to-program interface
PPMSG	program-to-program message format
PPT	program properties table
PR	permanently resident record
PRC	prime computer room agent set
PRDT	physical record (block) descriptor table
PRPQ	programming request for price quotation
PR/SM*	Processor Resource/Systems Manager*
PS	VTAM presentation services
PSPI	product sensitive programming interface
PSW	program status word
PTF	program temporary fix
PTT	Post Telephone and Telegraph Administration

PU	physical unit
PVC	permanent virtual circuit
QSAM	queued sequential access method
RACF*	resource access control facility
RB	request block
RBA	relative byte address
RCC	record code check
RCPL	routing control parameter list
RCR	resource control record
RCS	regional control center
RDB	Relational Database
RDBM	Relational Database Manager
REI	resource entry index
RLT	record locator table
RMF*	Resource Measurement Facility*
RO CRAS	receive-only computer room agent set
RON	record ordinal number
RPL	VTAM request parameter list
RPQ	request for price quotation
RSM	resume (SLC LCB)
RTM	recovery and termination management
RU	request unit
SAA*	Systems Application Architecture*
SAF	System Authorization Facility
SAL	system allocator list (TPF terminology)
SAM	sequential access method
SDLC	Synchronous Data Link Control
SDMF	standard data and message file
SDSF	System Display and Search Facility
SDWA	system diagnostic work area
SI	DBCS shift in
SITA**	Société Internationale de Télécommunications Aéronautiques
SLC	ATA/IATA synchronous link control
SLIP	serviceability level indication processing
SLN	symbolic line number
SLR	Service Level Reporter
SLU	secondary logical unit
SMP/E	System Modification Program Extended
SNA	Systems Network Architecture
SO	DBCS shift out
SON	system ordinal number
SQA	system queue area
SQL	Structured Query Language
SQLCA	SQL Communication Area
SQLDA	SQL Descriptor Area
SRB	service request block
SRG	statistical report generator
SRM	System Resource Manager
STC	system test compiler
STP	stop (SLC LCB)
STV	system test vehicle
SWB	service work block
SYN	character synchronization character
TA	terminal address

TAS	time available supervisor
TCB	task control block
TCID	terminal circuit identity
TCP/IP	Transmission Control Protocol / Internet Protocol
TI	time-initiated record
TOD	time of day
TPF	Transaction Processing Facility
TPF/APPC	Transaction Processing Facility/Advanced Program to Program Communications
TPF/DBR	Transaction Processing Facility/Data Base Reorganization
TPFDF	TPF Database Facility
TPF/MVS	Transaction Processing Facility/MVS (alternative name for ALCS V2)
TP_ID	transaction program identifier
TSI	transmission status indicator
TSN	transmission sequence number
TSO	time-sharing option
TSO/E	Time Sharing Option Extensions
TUT	test unit tape (sequential file)
UCB	unit control block
UCTF	Universal Communications Test Facility
VFA	virtual file access
VIPA	virtual IP address
VM	virtual machine
VM/CMS	virtual machine/conversational monitor system
VS	virtual storage
VSAM	virtual storage access method
VSE	Virtual Storage Extended
VSE/AF	Virtual Storage Extended/Advanced Function
VSE/VSAM	Virtual Storage Extended/Virtual Storage Access Method
VTAM*	Virtual Telecommunications Access Method (refers to VTAM)
VTOC	volume table of contents
WAS	WebSphere Application Server
WSF	Write Structured Field
WTTY	World Trade Teletypewriter
XMSG	XMSG message switching message format
XREF	ALCS cross referencing facility

Glossary

Notes:

1. Acronyms and abbreviations are listed separately from this Glossary. See “Acronyms and abbreviations” on page 45.
2. For an explanation of any term not defined here, see the IBM *Dictionary of Computing*.

A

AAA hold. See terminal hold.

abnormal end of task (abend). Termination of a task before its completion because of an error condition that cannot be resolved by recovery facilities while the task is executing.

access method services (AMS). A utility program that defines VSAM data sets (or files) and allocates space for them, converts indexed sequential data sets to key-sequenced data sets with indexes, modifies data set attributes in the catalog, facilitates data set portability between operating systems, creates backup copies of data sets and indexes, helps make inaccessible data sets accessible, and lists data set records and catalog entries.

activity control variable. A parameter that ALCS uses to control its workload. The system programmer defines activity control variables in the ALCS system configuration table generation.

Advanced Communications Function for the Network Control Program (ACF/NCP). An IBM licensed program that provides communication controller support for single-domain, multiple-domain, and interconnected network capability.

Advanced Program-to-Program Communications (APPC). A set of inter-program communication services that support cooperative transaction processing in an SNA network. APPC is the implementation, on a given system, of SNA’s logical unit type 6.2 (LU 6.2). See APPC component and APPC transaction scheduler.

Aeronautical Radio Incorporated (ARINC). An organization which provides communication facilities for use within the airline industry.

agent assembly area (AAA). A fixed-file record used by IPARS applications. One AAA record is associated with each terminal and holds data that needs to be kept beyond the life of an entry. For example, to collect information from more than one message.

agent set. Synonym for communication terminal.

agent set control unit (ASCU). Synonym for terminal interchange.

Airline Control Program (ACP). An earlier version of the IBM licensed program Transaction Processing Facility (TPF).

Airline Control System (ALCS). A transaction processing platform providing high performance, capacity, and availability, that runs specialized (typically airline) transaction processing applications.

Airline Control System/Multiple Virtual Storage/Extended Architecture (ALCS/MVS/XA). An ALCS release designed to run under an MVS/XA operating system.

Airline Control System Version 2 (ALCS V2). An ALCS release designed to run under a z/OS operating system.

Airline Control System/Virtual Storage Extended (ALCS/VSE). An ALCS release designed to run under a VSE/AF operating system.

airlines line control (ALC). A communication protocol particularly used by airlines.

Airlines Line Control Interconnection (ALCI). A feature of Network Control Program (NCP) that allows it to manage ALC networks in conjunction with a request for price quotation (RPQ) scanner for the IBM 3745 communication controller.

Airline X.25 (AX.25). A discipline conforming to the ATA/IATA AX.25 specification in the ATA/IATA publication *ATA/IATA Interline Communications Manual*, ATA/IATA document DOC.GEN 1840. AX.25 is based on X.25 and is intended for connecting airline computer systems to SITA or ARINC networks.

ALCS command. A command addressed to the ALCS system. All ALCS commands start with the letter Z (they are also called “Z messages”) and are 5 characters long.

These commands allow the operator to monitor and control ALCS. Many of them can only be entered from CRAS terminals. ALCS commands are called “functional messages” in TPF.

ALCS data collection file. A series of sequential data sets to which ALCS writes performance-related data for subsequent processing by the statistical report

generator or other utility program. See also data collection and statistical report generator.

ALCS diagnostic file. A series of sequential data sets to which the ALCS monitor writes all types of diagnostic data for subsequent processing by the diagnostic file processor.

ALCS diagnostic file processor. An offline utility, often called the “post processor”, that reads the ALCS diagnostic file and formats and prints the dump, trace, and system test vehicle (STV) data that it contains.

ALCS entry dispatcher. The ALCS online monitor's main work scheduler. Often called the “CPU loop”.

ALCS offline program. An ALCS program that runs as a separate MVS job (not under the control of the ALCS online monitor).

ALCS online monitor. The part of ALCS that performs the services for the ECB-controlled programs and controls their actions.

ALCS trace facility. An online facility that monitors the execution of application programs. When it meets a selected monitor-request macro, it interrupts processing and sends selected data to an ALCS display terminal, to the ALCS diagnostic file, or to the system macro trace block. See also instruction step.

The ALCS trace facility also controls tracing to the MVS generalized trace facility (GTF), for selected VTAM communication activity.

ALCS update log file. A series of sequential data sets in which the ALCS monitor records changes to the real-time database.

ALCS user file. A series of sequential data sets to which you may write all types of diagnostic data for subsequent processing by an offline processor. You write the data from an installation-wide monitor exit using the callable service UWSEQ.

allocatable pool. The ALCS record class that includes all records on the real-time database. Within this class, there is one record type for each DASD record size.

The allocatable pool class is special in that ALCS itself can dispense allocatable pool records and use them for other real-time database record classes. For example, all fixed-file records are also allocatable pool records (they have a special status of “in use for fixed file”).

When ALCS is using type 2 long-term pool dispense, ALCS satisfies requests for long-term pool by dispensing available allocatable pool records.

See DASD record, real-time database, record class, and record type.

alternate CRAS. A computer room agent set (CRAS) that is not Prime CRAS or receive only CRAS. See computer room agent set, Prime CRAS, and receive only CRAS.

alternate CRAS printer. A CRAS printer that is not receive only CRAS. See CRAS printer and receive only CRAS.

answerback. A positive acknowledgement (ACK) from an ALC printer.

APPC component. The component of MVS that is responsible for extending LU 6.2 and SAA CPI Communications services to applications running in any MVS address space. Includes APPC conversations and scheduling services.

APPC transaction scheduler. A program such as ALCS that is responsible for scheduling incoming work requests from cooperative transaction programs.

application plan. See DB2 application plan.

application. A group of associated application programs that carry out a specific function.

application global area. An area of storage in the ALCS address space containing application data that any entry can access.

The application global area is subdivided into keypointable and nonkeypointable records. Keypointable records are written to the database after an update; nonkeypointable records either never change, or are reinitialized when ALCS restarts.

C programs refer to global records and global fields within the application global area.

application program. A program that runs under the control of ALCS. See also ECB-controlled program.

application program load module. In ALCS, a load module that contains one or more application programs.

application queue. In message queuing with ALCS, any queue on which application programs put and get messages using MQI calls.

assign. Allocate a general sequential file to an entry. The `TOPNC` monitor-request macro (or equivalent C function) opens and allocates a general sequential file. The `TASNC` monitor-request macro (or equivalent C function) allocates a general sequential file that is already open but not assigned to an entry (it is reserved).

associated resource. Some ALCS commands generate output to a printer (for example, `ZDCOM` prints information about a communication resource). For this type of command the printed output goes to the

associated resource; that is, to a printer associated with the originating display. There is also a response to the originating display that includes information identifying the associated resource.

asynchronous trace. One mode of operation of the ALCS trace facility. Asynchronous trace is a conversational trace facility to interactively trace entries that do not originate from a specific terminal.

automatic storage block. A storage block that is attached to an entry, but is not attached at a storage level. An assembler program can use the ALASC monitor-request macro to obtain an automatic storage block and BACKC monitor-request macro to release it. C programs cannot obtain automatic storage blocks.

B

backward chain. The fourth fullword of a record stored on the ALCS database, part of the record header. See chaining of records.

When standard backward chaining is used, this field contains the file address of the previous record in the chain, except that the first record contains the file address of the last record in the chain. (If there is only one record, the backward chain field contains zeros.)

balanced path. A path where no single component (channel, DASD director or control unit, head of string, and internal path to the DASD device) is utilized beyond the limits appropriate to the required performance.

bar. In the MVS 64-bit address space, a virtual line called the bar marks the 2-gigabyte address. The bar separates storage below the 2-gigabyte address, called **below the bar**, from storage above the 2-gigabyte address, called **above the bar**.

BATAP. Type B application-to-application program

Binary Synchronous Communication (BSC). A form of telecommunication line control that uses a standard set of transmission control characters and control character sequences, for binary synchronous transmission of binary-coded data between stations.

bind. See DB2 bind

BIND. In SNA, a request to activate a session between two logical units (LUs). The BIND request is sent from a primary LU to a secondary LU. The secondary LU uses the BIND parameters to help determine whether it will respond positively or negatively to the BIND request.

binder. The program that replaces the linkage editor and batch loader programs that were provided with earlier versions of MVS.

BIND image. In SNA, the set of fields in a BIND request that contain the session parameters.

block. See storage block.

C

catastrophic. A type of system error that results in the termination of ALCS.

chain-chase. See Recoup.

chaining of records. One record can contain the file address of another (usually a pool-file record). The addressed record is said to be chained from the previous record. Chains of records can contain many pool-file records. See forward chain and backward chain.

class. See record class.

clearly differentiated programming interfaces (CDPI). A set of guidelines for developing and documenting product interfaces so that there is clear differentiation between interfaces intended for general programming use (GUPIs) and those intended for other specialized tasks.

close. Close a sequential file data set (MVS CLOSE macro) and deallocate it from ALCS. For general sequential files this is a function of the TCLSC monitor-request macro (or equivalent C function). ALCS automatically closes other sequential files at end-of-job.

command. See ALCS command.

command list (CLIST). A sequential list of commands, control statements, or both, that is assigned a name. When the name is invoked the commands in the list are executed.

commit. An operation that terminates a unit of recovery. Data that was changed is now consistent.

common entry point (CEP). A function in the Transaction Processing Facility Database Facility (TPFDF) product that provides common processing for all TPFDF macro calls issued by ALCS application programs. It also provides trace facilities for TPFDF macro calls.

Common Programming Interface – Communications (CPI-C). The communication element of IBM Systems Application Architecture (SAA). CPI-C provides a programming interface that allows program-to-program communication using the IBM SNA logical unit 6.2.

Common User Access. Guidelines for the dialog between a user and a workstation or terminal.

communication management configuration (CMC). A technique for configuring a network that allows for the consolidation of many network management functions for the entire network in a single host processor.

communication resource. A communication network component that has been defined to ALCS. These include each terminal on the network and other network components that ALCS controls directly (for example, SLC links). Resources can include, for example:

SNA LUs (including LU 6.1 links)
ALC terminals
SLC and WTTY links
Applications.

communication resource identifier (CRI). A 3-byte field that uniquely identifies an ALCS communication resource. It is equivalent to the LN/IA/TA in TPF and the LN/ARID in ALCS/VSE. ALCS generates a CRI for each resource.

communication resource name (CRN). A 1- to 8-character name that uniquely identifies an ALCS communication resource. For SNA LUs, it is the LU name. The system programmer defines the CRN for each resource in the ALCS communication generation.

communication resource ordinal. A unique number that ALCS associates with each communication resource. An installation can use the communication resource ordinal as a record ordinal for a particular fixed-file record type. This uniquely associates each communication resource with a single record.

For example, IPARS defines a fixed-file record type (#WAARI) for AAA records. Each communication resource has its own AAA record — the #WAARI record ordinal is the communication resource ordinal. See also record ordinal and agent assembly area.

compiler. A program that translates instructions written in a high level programming language into machine language.

computer room agent set (CRAS). An ALCS terminal that is authorized for the entry of restricted ALCS commands.

Prime CRAS is the primary terminal that controls the ALCS system. Receive Only CRAS (RO CRAS) is a designated printer or NetView operator identifier to which certain messages about system function and progress are sent.

configuration data set. (1) A data set that contains configuration data for ALCS. See also configuration-dependent table. (2) The ALCS record class that includes all records on the configuration data set. There is only one record type for this class. See record class and record type.

configuration-dependent table. A table, constructed by the ALCS generation process, which contains configuration-dependent data. Configuration-dependent tables are constructed as conventional MVS load modules. In ALCS V2, there are separate configuration-dependent tables for:

System data
DASD data
Sequential file data
Communication data
Application program data.

See also configuration data set.

control byte. The fourth byte of a record stored on the ALCS database, part of the record header. ALCS ignores this byte; some applications, however, make use of it.

control interval (CI). A fixed-length area of direct access storage in which VSAM stores records. The control interval is the unit of information that VSAM transmits to or from direct access storage.

control transfer. The process that the ALCS online monitor uses to create a new entry and to transfer control to an ECB-controlled program.

conversation_ID:. An 8-byte identifier, used in Get_Conversation calls, that uniquely identifies a conversation. APPC/MVS returns a conversation_ID on the CMINIT, ATBALLOC, and ATBGETC calls; a conversation_ID is required as input on subsequent APPC/MVS calls.

CPU loop. See ALCS entry dispatcher.

CRAS printer. A computer room agent set (CRAS) that is a printer terminal. See computer room agent set.

CRAS display. A computer room agent set (CRAS) that is a display terminal. See computer room agent set.

CRAS fallback. The automatic process that occurs when the Prime CRAS or receive only CRAS becomes unusable by which an alternate CRAS becomes Prime CRAS or receive only CRAS. See also Prime CRAS, receive only CRAS, and alternate CRAS.

create service. An ALCS service that enables an ALCS application program to create new entries for asynchronous processing. The new ECBs compete for system resources and, once created, are not dependent or connected in any way with the creating ECB.

cycling the system. The ALCS system can be run in one of four different system states. Altering the system state is called cycling the system. See SLC link for another use of the term “cycling”.

D

DASD record. A record stored on a direct access storage device (DASD). ALCS allows the same range of sizes for DASD records as it allows for storage blocks, except no size L0 DASD records exist.

data collection. An online function that collects data about selected activity in the system and sends it to the ALCS data collection file, if there is one, or to the ALCS diagnostic file. See also statistical report generator.

database request module (DBRM). A data set member created by the DB2 precompiler that contains information about SQL statements. DBRMs are used in the DB2 bind process. See DB2 bind.

data-collection area. An ECB area used by the ALCS online monitor for accumulating statistics about an entry.

data event control block (DECB). An ALCS control block, that may be acquired dynamically by an entry to provide a storage level and data level in addition to the 16 ECB levels. It is part of entry storage.

The ALCS DECB is independent of the MVS control block with the same name.

Data Facility Storage Management Subsystem (DFSMS*). An MVS operating environment that helps automate and centralize the management of storage. It provides the storage administrator with control over data class, management class, storage group, and automatic class selection routine definitions.

Data Facility Sort (DFSORT*). An MVS utility that manages sorting and merging of data.

data file. A sequential data set, created by the system test compiler (STC) or by the `ZDATA DUMP` command, that contains data to be loaded on to the real-time database. (An ALCS command `ZDATA LOAD` can be used to load data from a data file to the real-time database.) A data file created by STC is also called a “pilot” or “pilot tape”.

data level. An area in the ECB or a DECB used to hold the file address, and other information about a record. See ECB level and DECB level.

data record information library (DRIL). A data set used by the system test compiler (STC) to record the formats of data records on the real-time system. DRIL is used when creating data files.

DB2 application plan. The control structure produced during the bind process and used by DB2 to process SQL statements encountered during program execution. See DB2 bind.

DB2 bind. The process by which the output from the DB2 precompiler is converted to a usable control structure called a package or an application plan. During the process, access paths to the data are selected and some authorization checking is performed.

DB2 Call Attach Facility (CAF). An interface between DB2 and batch address spaces. CAF allows ALCS to access DB2.

DB2 for z/OS. An IBM licensed program that provides relational database services.

DB2 host variable. In an application program, an application variable referenced by embedded SQL statements.

DB2 package. Also called application package. An object containing a set of SQL statements that have been bound statically and that are available for processing. See DB2 bind.

DB2 package list. An ordered list of package names that may be used to extend an application plan.

DECB level. When an application program, running under ALCS, reads a record from a file, it must “own” a storage block in which to put the record. The address of the storage block may be held in an area of a DECB called a storage level.

Similarly, there is an area in a DECB used for holding the 8-byte file address, record ID, and record code check (RCC) of a record being used by an entry. This is a data level.

The storage level and data level in a DECB, used together, are called a DECB level.

See also ECB level.

diagnostic file. See ALCS diagnostic file.

dispatching priority. A number assigned to tasks, used to determine the order in which they use the processing unit in a multitasking situation.

dispense (a pool-file record). To allocate a long-term or short-term pool-file record to a particular entry. ALCS performs this action when requested by an application program. See release a pool-file record.

double-byte character set. A set of characters in which each character is represented by 2 bytes. Languages such as Japanese, Chinese, and Korean, which contain more symbols than can be represented by 256 code points, require double-byte character sets.

Because each character requires 2 bytes, entering, displaying, and printing DBCS characters requires hardware and supporting software that are DBCS-capable.

duplex. A communication link on which data can be sent and received at the same time. Synonymous with full duplex. Communication in only one direction at a time is called “half-duplex”. Contrast with simplex transmission.

duplex database. Synonym for duplicated database.

duplicated database. A database where each data set is a mirrored pair. In ALCS, you can achieve this using either ALCS facilities or DASD controller facilities (such as the IBM 3990 dual copy facility). See mirrored pair.

dynamic program linkage. Program linkage where the connection between the calling and called program is established during the execution of the calling program. In ALCS dynamic program linkage, the connection is established by the ALCS ENTER/BACK services. Contrast with static program linkage.

dynamic SQL. SQL statements that are prepared and executed within an application program while the program is executing. In dynamic SQL, the SQL source is contained in host language variables rather than being coded into the application program. The SQL statement can change several times during the application program's execution. Contrast with embedded SQL.

E

ECB-controlled program. A program that runs under the control of an entry control block (ECB). These programs can be application programs or programs that are part of ALCS, for example the ALCS programs that process operator commands (Z messages). ECB-controlled programs are known as E-type programs in TPF.

ECB level. When an application program, running under ALCS, reads a record from file, it must “own” a storage block in which to put the record. The address of the storage block may be held in an area of the ECB called a storage level.

There are 16 storage levels in the ECB. A storage block with its address in slot zero in the ECB is said to be attached on level zero.

Similarly, there are 16 areas in the ECB that may be used for holding the 4-byte file addresses, record ID, and record code check (RCC) of records being used by an entry. These are the 16 data levels.

Storage levels and data levels, used together, are called ECB levels.

See also DECB level.

embedded SQL. Also called static SQL. SQL statements that are embedded within an application program and are prepared during the program preparation process before the program is executed. After it is prepared, the statement itself does not change (although values of host variables specified within the statement can change). Contrast with dynamic SQL.

Emulation Program/Virtual Storage (EP/VS). A component of NCP/VS that ALCS V2 uses to access SLC networks.

ENTER/BACK. The general term for the application program linkage mechanism provided by ALCS.

entry. The basic work scheduling unit of ALCS. An entry is represented by its associated entry control block (ECB). It exists either until a program that is processing that entry issues an `EXITC` monitor-request macro (or equivalent C function), or until it is purged from the system. An entry is created for each input message, as well as for certain purposes unrelated to transactions. One transaction can therefore generate several entries.

entry control block (ECB). A control block that represents a single entry during its life in the system.

entry dispatcher. See ALCS entry dispatcher.

entry macro trace block. There is a macro trace block for each entry. Each time an entry executes a monitor-request macro (or a corresponding C function), ALCS records information in the macro trace block for the entry.

This information includes the macro request code, the name of the program that issued the macro, and the displacement in the program. The ALCS diagnostic file processor formats and prints these macro trace blocks in ALCS system error dumps.

See also system macro trace block.

entry storage. The storage associated with an entry. It includes the ECB for the entry, storage blocks that are attached to the ECB or DECBs, storage blocks that are detached from the ECB or DECBs, automatic storage blocks, and DECBs. It also includes heap storage (for high-level language or assembler language programs) and stack storage (for high-level language programs).

equate. Informal term for an assignment instruction in assembler languages.

error index byte (EIB). See SLC error index byte.

extended buffer. A storage area above 2 GB used for large messages.

extended message format. For input and output messages, a message format which includes a 4-byte field for the message length.

Execute Channel Program (EXCP). An MVS macro used by ALCS V2 to interface to I/O subsystems for SLC support.

F

fetch access. Access which only involves reading (not writing). Compare with store access.

file address. 4-byte (8 hexadecimal digits) value or 8-byte value in 4x4 format (low order 4-bytes contain a 4-byte file address, high order 4 bytes contain hexadecimal zeros) that uniquely identifies an ALCS record on DASD. `FIND/FILE` services use the file address when reading or writing DASD records. See fixed file and pool file.

file address compute routine (FACE). An ALCS routine, called by a monitor-request macro (or equivalent C function) that calculates the file address of a fixed-file record. The application program provides the FACE routine with the fixed-file record type and the record ordinal number. FACE returns the 4-byte file address.

There is also an `FAC8C` monitor-request macro (or equivalent C function), that will return an 8-byte file address in 4x4 format.

FIND/FILE. The general term for the DASD I/O services that ALCS provides.

fixed file. An ALCS record class – one of the classes that reside on the real-time database. All fixed-file records are also allocatable pool records (they have a special status of "in use for fixed file").

Within this class there are two record types reserved for use by ALCS itself (#KPTRI and #CPRCR). There can also be installation-defined fixed-file record types.

Each fixed-file record type is analogous to a relative file. Applications access fixed-file records by specifying the fixed-file record type and the record ordinal number. Note however that fixed-file records are not physically organized as relative files (logically adjacent records are not necessarily physically adjacent).

See real-time database, record class, and record type. See also system fixed file. Contrast with pool file.

fixed-file record. One of the two major types of record in the real-time database (the other is a pool-file record). When the number of records of a particular kind will not vary, the system programmer can define a fixed file record type for these records. ALCS application programs accessing fixed-file records use the `ENTRC` monitor-request macro to invoke the 4-byte file address compute routine (FACE or FACS) or use the `FAC8C` monitor-request macro to compute an 8-byte file address. The equivalent C functions are `face` or `facs` or `tpf_fac8c`.

fixed-file record type. (Known in TPF as FACE ID.) The symbol, by convention starting with a hash sign (#)[1] which identifies a particular group of fixed-file records. It is called the fixed-file record type symbol. The equated value of this symbol (called the fixed-file record type value) also identifies the fixed-file record type.

forward chain. The third fullword of a record stored on the ALCS database (part of the record header). When standard forward chaining is used, this field contains the file address of the next record in the chain, except that the last (or only) record contains binary zeros.

full-duplex. Deprecated term for duplex.

functional message. See ALCS command.

G

general data set (GDS). The same as a general file, but accessed by different macros or C functions in ALCS programs.

general file. (1) A DASD data set (VSAM cluster) that is used to communicate data between offline utility programs and the online system. General files are not part of the real-time database. (2) The ALCS record class that includes all records on the general files and general data sets. Each general file and general data set is a separate record type within this class. See record class and record type.

general file record. A record on a general file.

generalized trace facility (GTF). An MVS trace facility. See also ALCS trace facility.

general sequential file. A class of sequential data set that is for input or output. ALCS application programs must have exclusive access to a general sequential file before they can read or write to it. See also real-time sequential file.

general tape. TPF term for a general sequential file.

general-use programming interface (GUPI). An interface intended for general use in customer-written applications.

get file storage (GFS). The general term for the pool file dispense mechanisms that ALCS provides.

---

[1] This character might appear differently on your equipment. It is the character represented by hexadecimal 7B.

global area. See application global area.

global resource serialization. The process of controlling access of entries to a global resource so as to protect the integrity of the resource.

H

half-duplex. A communication link that allows transmission in one direction at a time. Contrast with duplex.

halt. (1) The ALCS state when it is terminated. (2) The action of terminating ALCS.

heap. An area of storage that a compiler uses to satisfy requests for storage from a high-level language (for example, `calloc` or `malloc` C functions). ALCS provides separate heaps for each entry (if needed). The heap is part of entry storage. Assembler language programs may also obtain or release heap storage using the `CALOC`, `MALOC`, `RALOC`, and `FREEC` monitor-request macros.

High Level Assembler (HLASM). A functional replacement for Assembler H Version 2. HLASM contains new facilities for improving programmer productivity and simplifying assembler language program development and maintenance.

high-level language (HLL). A programming language such as C or COBOL.

high-level language (HLL) storage unit. Alternative name for a type 2 storage unit. See storage unit.

high-level network (HLN). A network that provides transmission services between transaction processing systems (for example, ALCS) and terminals. Strictly, the term “high-level network” applies to a network that connects to transaction processing systems using SLC. But in ALCS publications, this term is also used for a network that connects by using AX.25 or MATIP.

high-level network designator (HLD). The entry or exit point of a block in a high-level network. For SLC networks, it is the SLC address of a switching center that is part of a high-level network. It comprises two bytes in the 7-bit transmission code used by SLC.

HLN entry address (HEN). The high-level designator of the switching center where a block enters a high-level network.

HLN exit address (HEX). The high-level designator of the switching center where a block leaves a high-level network.

hold. A facility that allows multiple entries to share data, and to serialize access to the data. The data can be a database record, or any named data resource. This facility can be used to serialize conflicting processes. See also record hold and resource hold.

host variable. See DB2 host variable

I

information block. See SLC link data block.

initial storage allocation (ISA). An area of storage acquired at initial entry to a high-level language program. ALCS provides a separate ISA for each entry (if required). The ISA is part of entry storage.

initiation queue. In message queuing, a local queue on which the queue manager puts trigger messages. You can define an initiation queue to ALCS, in order to start an ALCS application automatically when a trigger message is put on the queue. See trigger message.

input/output control block (IOCB). A control block that represents an ALCS internal “task”. For example, ALCS uses an IOCB to process a DASD I/O request.

input queue. In message queuing with ALCS, you can define a local queue to ALCS in order to start an ALCS application automatically when a message is put on that queue. ALCS expects messages on the input queue to be in PPMSG message format. See PPMSG.

installation-wide exit. The means specifically described in an IBM software product’s documentation by which an IBM software product may be modified by a customer’s system programmers to change or extend the functions of the IBM software product. Such modifications consist of exit routines written to replace an existing module of an IBM software product, or to add one or more modules or subroutines to an IBM software product for the purpose of modifying (including extending) the functions of the IBM software product. Contrast with user exit.

instruction step. One mode of operation of the ALCS trace facility. Instruction step is a conversational trace facility that stops the traced application program before the execution of each processor instruction.

Interactive System Productivity Facility (ISPF). An IBM licensed program that serves as a full-screen editor and dialog manager. ISPF provides a means of generating standard screen panels and interactive dialog between the application programmer and terminal user.

interchange address (IA). In ALC, the 1-byte address of a terminal interchange. Different terminal interchanges connected to the same ALC link have different interchange addresses. Different terminal interchanges connected to different ALC links can have

the same interchange address. See also terminal interchange

International Programmed Airlines Reservation System (IPARS). A set of applications for airline use. The principal functions are reservations and message switching.

IPARS for ALCS. The ALCS shipment includes IPARS as a sample application, and installation verification aid for ALCS.

K

KCN. Abbreviation for an SLC channel number. See SLC channel.

keypointable. See application global area.

keypoint B (CTKB). A record that contains dynamic system information that ALCS writes to DASD when it is updated so that ALCS can restart from its latest status.

L

Language Environment*. A common run-time environment and common run-time services for z/OS high level language compilers.

level. See ECB level.

line number (LN). (1) In ALC, the 1-byte address of an ALC link. Different links connected to the same communication controller have different line numbers. Different links connected to different communication controllers can have the same line number.
(2) Synonym for symbolic line number.

Link Control — Airline (LICRA). The name of a programming request for price quotation (PRPQ) to the IBM 3705 Emulation Program (EP/VS). This modifies EP/VS to support SLC networks.

link control block (LCB). See SLC link control block.

link data block (LDB). See SLC link data block.

link trace. See SLC link trace.

local DXCREI index (LDI). The first byte of a communication resource indicator (CRI).

local queue. In message queuing, a queue that belongs to the local queue manager. A local queue can contain a list of messages waiting to be processed. Contrast with remote queue.

lock. A serialization mechanism whereby a resource is restricted for use by the holder of the lock. See also hold.

log. See ALCS update log.

logging. The process of writing copies of altered database records to a sequential file. This is the method used to provide an up-to-date copy of the database should the system fail and the database have to be restored. The database records are logged to the ALCS update log file.

logical end-point identifier (LEID). In NEF2 and ALCI environments, a 3-byte identifier assigned to an ALC terminal.

logical unit type 6.2 (LU 6.2). The SNA logical unit type that supports general communication between programs in a distributed processing environment; the SNA logical unit type on which Common Programming Interface – Communications (CPI-C) is built.

log in. TPF term for establishing routing between a terminal and an application.

log on. Establish a session between an SNA terminal and an application such as ALCS. See also routing.

logon mode. In VTAM, a set of predefined session parameters that can be sent in a BIND request. When a set is defined, a logon mode name is associated with the set.

logon mode table. In VTAM, a table containing several predefined session parameter sets, each with its own logon mode name.

long message transmitter (LMT). A part of the IPARS application that is responsible for blocking and queuing printer messages for output. Also called XLMT.

long-term pool. An ALCS record class – one of the classes that reside on the real-time database. Within this class, there is one record type for each DASD record size. All long-term pool-file records are also allocatable pool records. ALCS application programs can use long-term pool records for long-lived or high-integrity data. See pool file, real-time database, record class, and record type.

L0, L1, L2, L3, ..., L8. Assembler symbols (and defined values in C) for the storage block sizes and record sizes that ALCS supports. See DASD record and storage block size.

M

macro trace block. See entry macro trace block and system macro trace block.

Mapping of Airline Traffic over IP (MATIP). A protocol for transporting traditional airline messages over an IP (Internet Protocol) network. Internet RFC (Request for Comments) number 2351 describes the MATIP protocol.

MBI exhaustion. The condition of an SLC link when a sender cannot transmit another message because all 7 SLC message labels are already "in use"; that is, the sender must wait for acknowledgement of a message so that it can reuse the corresponding message label. See also SLC link, SLC message label, and SLC message block indicator.

message. For terminals with an Enter key, an input message is the data that is sent to the host when the Enter key is hit. A response message is the data that is returned to the terminal. WTTY messages have special "start/end of message" character sequences. One or more input and output message pairs make up a transaction.

message block indicator. See SLC message block indicator.

message label. See SLC message label.

Message Queue Interface (MQI). The programming interface provided by the IBM WebSphere MQ message queue managers. This programming interface allows application programs to access message queuing services.

message queue manager. See queue manager.

message queuing. A programming technique in which each program within an application communicates with the other programs by putting messages on queues. This enables asynchronous communication between processes that may not be simultaneously active, or for which no data link is active. The message queuing service can assure subsequent delivery to the target application.

message switching. An application that routes messages by receiving, storing, and forwarding complete messages. IPARS for ALCS includes a message switching application for messages that conform to ATA/IATA industry standards for interline communication *ATA/IATA Interline Communications Manual,* DOC.GEN/1840.

mirrored pair. Two units that contain the same data and are referred to by the system as one entity.

monitor-request macro. Assembler language macro provided with ALCS, corresponding to TPF "SVC-type" or "control program" macros. Application programs use these macros to request services from the online monitor.

MQ Bridge. The ALCS MQ Bridge allows application programs to send and receive messages using WebSphere MQ for z/OS queues, without the need to code MQ calls in those programs. The MQ Bridge installation-wide monitor exits USRMQB0, USRMQB1, USRMQB2, and USRMQB3 allow you to customize the behaviour of the MQ Bridge to suit your applications.

MQSeries*. A previous name for WebSphere MQ.

multibyte character. A mixture of single-byte characters from a single-byte character set and double-byte characters from a double-byte character set.

multiblock message. In SLC, a message that is transmitted in more than one link data block. See link data block.

Multiple Virtual Storage/Data Facility Product (MVS/DFP*). An MVS licensed program that isolates applications from storage devices, storage management, and storage device hierarchy management.

Multisystem Networking Facility (MSNF). An optional feature of VTAM that permits these access methods, together with NCP, to control a multiple-domain network.

N

namelist. In message queuing, a namelist is an object that contains a list of other objects.

native file address. For migration purposes ALCS allows two or more file addresses to refer to the same database or general file record. The file address that ALCS uses internally is called the native file address.

NCP Packet Switching Interface (NPSI). An IBM licensed program that allows communication with X.25 lines.

NetView*. A family of IBM licensed programs for the control of communication networks.

NetView operator identifier (NetView operator ID). A 1- to 8-character name that identifies a NetView operator.

NetView program. An IBM licensed program used to monitor a network, manage it, and diagnose network problems.

NetView resource. A NetView operator ID which identifies one of the following:

- A NetView operator logged on to a terminal.
- A NetView operator ID automation task. One of these tasks is used by ALCS to route RO CRAS messages to the NetView Status Monitor Log (STATMON).

network control block (NCB). A special type of message, used for communication between a transaction processing system and a high-level network (HLN). For example, an HLN can use an NCB to transmit information about the network to a transaction processing system.

For a network that connects using SLC, an NCB is an SLC link data block (LDB). Indicators in the LDB differentiate NCBs from other messages.

For a network that connects using AX.25, NCBs are transmitted across a dedicated permanent virtual circuit (PVC).

Network Control Program (NCP). An IBM licensed program resident in an IBM 37xx Communication Controller that controls attached lines and terminals, performs error recovery, and routes data through the network.

Network Control Program Packet Switching Interface (NPSI). An IBM licensed program that provides a bridge between X.25 and SNA.

Network Control Program/Virtual Storage (NCP/VS). An IBM licensed program. ALCS V2 uses the EP/VS component of NCP/VS to access SLC networks.

Network Extension Facility (NEF). The name of a programming request for price quotation (PRPQ P09021) that allows management of ALC networks by NCP; now largely superseded by ALCI.

Network Terminal Option (NTO). An IBM licensed program that converts start-stop terminal device communication protocols and commands into SNA and VTAM communication protocols and commands. ALCS uses NTO to support World Trade Teletypewriter (WTTY).

O

object. In message queuing, objects define the attributes of queue managers, queues, process definitions, and namelists.

offline. A function or process that runs independently of the ALCS online monitor. For example, the ALCS diagnostic file processor is an offline function. See also ALCS offline program.

online. A function or process that is part of the ALCS online monitor, or runs under its control. For example, all ALCS commands are online functions. See also ALCS online monitor.

open. Allocate a sequential file data set to ALCS and open it (MVS `OPEN` macro). For general sequential files this is a function of the `TOPNC` monitor-request macro (or equivalent C function). ALCS automatically opens other sequential files during restart.

optimized local adapters (OLA) for WebSphere Application Server for z/OS (WAS). Built-in, high-speed, bi-directional adapters for calls between WebSphere Application Server for z/OS and ALCS in another address space on the same z/OS image. OLA allows ALCS customers to support an efficient integration of newer Java-based applications with ALCS-based applications. A set of callable services can be used by ALCS assembler or C/C++ programs for exchanging data with applications running in WebSphere Application Server for z/OS. For more information on the callable services (with names of the form BBOA1*xxx*) see the IBM Information Center for WebSphere Application Server - Network Deployment (z/OS) and search for BBOA1. You can use the USRWAS1 installation-wide monitor to verify the caller's authority and to identify input and output messages.

operator command. See ALCS command. Can also refer to non-ALCS commands, for example, MVS or VTAM commands.

ordinal. See communication resource ordinal and record ordinal.

P

package. See DB2 package

package list. See DB2 package list

padded ALC. A transmission code that adds one or more bits to the 6-bit airline line control (ALC) transmission code so that each ALC character occupies one character position in a protocol that uses 7- or 8-bit transmission codes. See also airlines line control.

padded SABRE. Synonym for padded ALC.

passenger name record (PNR). A type of record commonly used in reservation systems. It contains all the recorded information about an individual passenger.

path. The set of components providing a connection between a processor complex and an I/O device. For example, the path for an IBM 3390 DASD volume might include the channel, ESCON Director, 3990 Storage Path, 3390 Device Adapter, and 3390 internal connection. The specific components used in a

particular path are dynamic and may change from one I/O request to the next. See balanced path.

pathlength. The number of machine instructions needed to process a message from the time it is received until the reponse is sent to the communication facilities.

performance monitor. An online function that collects performance data and stores it in records on the ALCS real-time database. It can produce online performance reports based on current data and historical data.

pilot. See data file.

pool directory update (PDU). A facility of TPF that recovers long-term pool file addresses without running Recoup. PDU identifies and makes available all long-term pool-file records that have been released.

pool file. Short-term pool, long-term pool, and allocatable pool. Within each pool file class, there is one record type for each record size; for example, short-term pool includes the record type L1STPOOL (size L1 short-term pool records).

Each pool-file record type contains some records that are in-use and some that are available. There is a dispense function that selects an available record, changes its status to in-use, and returns the file address. Also, there is a release function that takes the file address of an in-use pool-file record and changes the record status to available.

To use a pool-file record, a program must:

1. Request the dispense function. This returns the file address of a record. Note that the record contents are, at this stage, unpredictable.
2. Write the initial record contents, using the file address returned by step 1.
3. Save the file address returned by step 1.
4. Read and write the record to access and update the information as required. These reads and writes use the file address saved in step 3.

When the information in the record is no longer required, a program must:

5. Delete (clear to zeros) the saved copy of the file address (see step 3).
6. Request the release function.

See also record class. Contrast with fixed file.

pool file directory record (PFDR). The ALCS pool file management routine keeps a directory for each size (L1, L2, ...L8) of short-term pool file records and long-term pool-file records. It keeps these directories in pool file directory records.

pool-file record. ALCS application programs access pool-file records with file addresses similar to those for fixed-file records. To obtain a pool-file record, an application program uses a monitor-request macro (or equivalent C function) that specifies a 2-byte record ID or a pool-file record type.

When the data in a pool-file record is no longer required, the application uses a monitor-request macro (or equivalent C function) to release the record for reuse. See pool file.

pool-file record identifier (record ID). The record ID of a pool-file record. On get file requests (using the `GETFC` monitor-request macro or equivalent C function) the program specifies the pool-file record ID. This identifies whether the pool-file record is a short-term or long-term pool-file record and also determines the record size (L1, L2, ...L8). (Coding the 2-byte record IDs, and the corresponding pool-file record sizes and types, is part of the ALCS generation procedure.) See also record ID qualifier.

pool-file record type. Each collection of short-term and long-term pool-file records of a particular record size (identified by the symbols L1, L2, ..., L8) is a different record type. Each pool-file record type has a different name. For short-term pool-file records, this is L*n*STPOOL, where L*n* is the record size symbol. For long-term pool-file records the name is L*n*LTPOOL.

post processor. See ALCS diagnostic file processor.

PPMSG. ALCS program-to-program message format, used by the ALCS message router to send and receive messages on a message routing path to another system. In PPMSG message format, the routing control parameter list (RCPL) precedes the message text.

primary action code. The first character of any input message. The primary action code Z is reserved for ALCS commands. See secondary action code.

Prime CRAS. The primary display terminal, or NetView ID, that controls the ALCS system. See also computer room agent set (CRAS).

process definition object. In message queuing, an object that contains the definition of a message queuing application. For example, a queue manager uses the definition when it works with trigger messages.

product sensitive programming interface (PSPI). An interface intended for use in customer-written programs for specialized purpose only, such as diagnosing, modifying, monitoring, repairing, tailoring or tuning of ALCS. Programs using this interface may need to be changed in order to run with new product releases or versions, or as a result of service.

program linkage. Mechanism for passing control between separate portions of the application program. See dynamic program linkage and static program linkage.

program nesting level. One of 32 ECB areas used by the ENTER/BACK mechanism for saving return control data.

program-to-program interface. In NetView, a facility that allows user programs to send data to, or receive data from, other user programs. It also allows system and application programs to send alerts to the NetView hardware monitor.

P.1024. A SITA implementation of SLC. See SLC.

P.1124. A SITA implementation of SLC. See SLC.

P.1024A. The SITA implementation of airline line control (ALC).

Q

queue manager. A system program that provides queuing services to applications. It provides an application programming interface so that programs can access messages on the queues that the queue manager owns. WebSphere MQ for z/OS is an example of a queue manager.

R

real-time database. The database to which ALCS must have permanent read and write access. As an ALCS generation option, the real-time database can be duplicated in order to minimize the effects of a DASD failure.

real-time sequential file. A sequential data set used only for output. ALCS application programs can write to any real-time sequential file without requiring exclusive access to the data set. See also general sequential file.

real-time tape. TPF term for a real-time sequential file.

receive only (RO). The function of a communication terminal that can receive but not send data. An example is a printer that does not have a keyboard.

receive only CRAS. A printer terminal (or NetView operator ID) that ALCS uses to direct status messages. Commonly known as RO CRAS.

record. A set of data treated as a unit.

record class. The first (highest) level categorization of ALCS DASD records. ALCS defines the following record classes:

Allocatable pool
Application fixed file
Configuration data set
General file
Long-term pool
Short-term pool
System fixed file.

See also record type and record ordinal.

record code check (RCC). The third byte of any record stored in the ALCS database. It is part of the record header.

The RCC field is intended to help detect the incorrect chaining of records which have the same record ID. This is particularly useful for passenger name records (PNRs), of which there are often hundreds of thousands. A mismatch in RCC values shows that the chain is broken, probably as a result of an application program releasing a record too soon. (A false match cannot be excluded, but the RCC should give early warning of a chaining problem.)

record header. A standard format for the first 16 bytes of a record stored on the ALCS database. It contains the following fields:

Record ID
Record code check
Control byte
Application program name
Forward chain
Backward chain.

Not all records contain forward chains and backward chains. Some applications extend the record header by including extra fields. TPFDF uses an extended record header.

record hold. A type of hold that applies to DASD records. Applications that update records can use record hold to prevent simultaneous updates. See also resource hold.

record identifier (record ID). The first two bytes of a record stored on the ALCS database, part of the record header.

The record ID should always be used to indicate the nature of the data in the record. For example, airlines reservations applications conventionally store passenger name records (PNRs) as long-term pool-file records with a record ID of `'PR'`.

When application programs read such records, they can (optionally) request ALCS to check that the record ID matches that which the application program expects.

When application programs request ALCS to dispense pool file records, ALCS uses the record ID to select an appropriate long-term or short-term pool-file record of the requested record size (L1, L2,...,L8). See also record ID qualifier.

record ID qualifier. A number 0 through 9 that differentiates between record types that have the same record ID.

For compatibility with previous implementations of the record ID qualifier, ALCS also accepts the character qualifiers P and O. P (primary) is equivalent to 0, and O (overflow) is equivalent to 1.

record ordinal. The relative record number within a record type. See record class and record type.

record size. See DASD record.

record type. The second level categorization of ALCS DASD records. Within any one record class, the records are categorized into one or more record types. See also record type number, record type symbol, record class and record ordinal.

record type number. A number that identifies a record type.

record type symbol. The character string that identifies a fixed-file record type (#*xxxxx*), a long-term pool-file record type (LsLTPOOL), a short-term pool-file record type (LsSTPOOL), or a general file (GF-*nnn*). The value of the record type symbol is the record type number.

Recoup. A real-time database validation routine which runs online in the ALCS system. (Note that, while the Recoup routines of TPF consist of a number of phases, some online and some offline, the ALCS Recoup is a single online phase that runs, without operator intervention, in any system state.)

Recoup reads selected fixed-file records in the database, and then follows up all chains of pool-file records in the database, noting that these records are in use and giving a warning of any that have been corrupted or released. It then updates the pool file directory records (PFDRs) to show the status of all records.

The ALCS pool file dispense procedure identifies records not in a chain (and so apparently available for reuse) that have not been released.

recoup descriptors. These describe the structure of the entire real-time database.

reentrant. The attribute of a program or routine that allows the same copy of the program or routine to be used concurrently by two or more tasks. All ALCS application programs must be reentrant.

relational database. A database that is in accordance with the relational model of data. The database is perceived as a set of tables, relationships are represented by values in tables, and data is retrieved by specifying a result table that can be derived from one or more base tables.

release (a pool-file record). To make available a long-term or short-term pool-file record so that it can be subsequently dispensed. An application program requests the release action. See dispense a pool-file record.

release file storage (RFS). The general term for the pool-file release mechanisms that ALCS provides.

remote queue. In message queuing, a queue that belongs to a remote queue manager. Programs can put messages on remote queues, but they cannot get messages from remote queues. Contrast with local queue.

remote terminal trace. One mode of operation of the ALCS trace facility. Remote terminal trace is a conversational trace facility to interactively trace entries from a terminal other than your own.

reservations. An online application which is used to keep track of seat inventories, flight schedules, and other related information. The reservation system is designed to maintain up-to-date data and to respond within seconds or less to inquiries from ticket agents at locations remote from the computing system.

IPARS for ALCS includes a sample reservations application for airlines.

reserve. Unassign a general sequential file from an entry but leave the file open, so that another (or the same) entry can assign it. Application programs can use the `TRSVC` monitor-request macro (or equivalent C function) to perform this action.

resource. Any facility of a computing system or operating system required by a job or task, and including main storage, input/output devices, processing unit, data sets, and control or processing programs. See also communication resource.

resource entry index (REI). The second and third bytes of a communication resource identifier (CRI).

resource hold. A type of hold that can apply to any type of resource. Applications can define resources according to their requirements, and identify them to ALCS using a unique name. See also record hold.

RO CRAS. See receive only CRAS.

rollback. An operation that reverses all the changes made during the current unit of recovery. After the operation is complete, a new unit of recovery begins.

routing. The connection between a communication resource connected to ALCS (typically a terminal on an

SNA or non-SNA network) and an application (running under ALCS or another system). Also sometimes called "logging in", but this must be distinguished from logging on, which establishes the SNA connection (session) between the terminal and ALCS.

routing control parameter list (RCPL). A set of information about the origin, destination, and characteristics of a message. With each input message, ALCS provides an RCPL in the ECB. An output message that is sent using the ROUTC (routc) service also has an RCPL associated with it.

S

scroll. To move a display image vertically or horizontally to view data that otherwise cannot be observed within the boundaries of the display screen.

secondary action code. The second character of an ALCS command. (ALCS commands are made up of 5 characters: Z followed by a secondary action code.) See primary action code.

sequential file. A file in which records are processed in the order in which they are entered and stored in the file. See general sequential file and real-time sequential file.

serialization. A service that prevents parallel or interleaved execution of two or more processes by forcing the processes to execute serially.

For example, two programs can read the same data item, apply different updates, and then write the data item. Serialization ensures that the first program to start the process (read the item) completes the process (writes the updated item) before the second program can start the process – the second program applies its update to the data item which already contains the first update. Without serialization, both programs can start the process (read the item) before either completes the process (writes the updated item) – the second write destroys the first update. See also assign, lock, and hold.

Serviceability Level Indicator Processing (SLIP). An MVS operator command which acts as a problem determination aid.

short-term pool. An ALCS record class – one of the classes that resides on the real-time database. Within this class, there is one record type for each DASD record size. All short-term pool-file records are also allocatable pool records (they have a special status of "in use for short-term pool"). ALCS application programs can use short-term pool records for short-lived low-integrity data. See pool file, real-time database, record class, and record type.

simplex transmission. Data transmission in one direction only. See also duplex and half-duplex.

sine in/out. Those applications that provide different functions to different end users of the same application can require the user to sine in[2] to the specific functions they require. The sine-in message can, for example, include an authorization code.

single-block message. In SLC, a message that is transmitted in one link data block. See link data block.

single-phase commit. A method in which a program can commit updates to a message queue or relational database without coordinating those updates with updates the program has made to resources controlled by another resource manager. Contrast with two-phase commit.

SLC. See synchronous link control.

SLC channel. A duplex telecommunication line using ATA/IATA SLC protocol. There can be from 1 to 7 channels on an SLC link.

SLC error index byte (EIB). A 1-byte field generated by Line Control – Airline (LICRA) and transferred to ALCS with each incoming link control block and link data block. Certain errors cause LICRA to set on certain bits of the EIB. See also Link Control — Airline (LICRA).

SLC information block. Synonym for SLC link data block.

SLC link. A processor-to-processor or processor-to-HLN connection. ALCS supports up to 255 SLC links in an SLC network.

An SLC link that is in the process of an open, close, start, or stop function is said to be "cycling".

SLC link control block (LCB). A 4-byte data item transmitted across an SLC link to control communications over the link. LCBs are used, for example, to confirm that a link data block (LDB) has arrived, to request retransmission of an LDB, and so on.

SLC link data block (LDB). A data item, transmitted across an SLC link, that contains a message or part of a message. One LDB can contain a maximum of 240 message characters, messages longer than this must

[2] This spelling is established in the airline industry.

be split and transmitted in multiple LDBs. Synonymous with SLC information block.

SLC link trace. A function that provides a record of SLC communication activity. It can either display the information in real time or write it to a diagnostic file for offline processing, or both. Its purpose is like that of an NCP line trace, but for the SLC protocol.

SLC message block indicator (MBI). A 1-byte field in the SLC link data block that contains the SLC message label and the block number. A multiblock message is transmitted in a sequence of up to 16 link data blocks with block numbers 1, 2, 3, ... 16. See also multiblock message, SLC link data block, and SLC message label.

SLC message label. A number in the range 0 through 7, excluding 1. In P.1024, consecutive multiblock messages are assigned SLC message labels in the sequence: 0, 2, 3, ... 6, 7, 0, 2, and so on. In P.1124, single-block messages are (optionally) also included in the sequence. See also P.1024, P.1124 and SLC message block indicator.

SLC transmission status indicator (TSI). A 1-byte field in the SLC link data block that contains the SLC transmission sequence number. See also SLC transmission sequence number.

SLC transmission sequence number (TSN). A number in the range 1 through 31. Consecutive SLC link data blocks transmitted in one direction on one SLC channel are assigned TSNs in the sequence: 1, 2, 3, ... 30, 31, 1, 2, and so on. See also SLC link data block, SLC channel, and SLC transmission status indicator.

SLC Type A traffic. See Type A traffic.

SLC Type B traffic. See Type B traffic.

Société Internationale de Télécommunications Aéronautiques (SITA). An international organization which provides communication facilities for use within the airline industry.

SQL Communication Area (SQLCA). A structure used to provide an application program with information about the execution of its SQL statements.

SQL Descriptor Area (SQLDA). A structure that describes input variables, output variables, or the columns of a result table used in the execution of manipulative SQL statements.

stack. An area of storage that a compiler uses to allocate variables defined in a high-level language. ALCS provides separate stacks for each entry (if needed). The stack is part of entry storage.

standard message format. For input and output messages, a message format which includes a 2-byte field for the message length.

standby. The state of ALCS after it has been initialized but before it has been started. Standby is not considered one of the system states.

static program linkage. Program linkage where the connection between the calling and called program is established before the execution of the program. The connection is established by the assembler, compiler, prelinker, or linkage editor. Static program linkage does not invoke ALCS monitor services. See also dynamic program linkage.

static SQL. See embedded SQL.

statistical report generator (SRG). An offline ALCS utility that is a performance monitoring tool. It takes the data written to the ALCS data collection or diagnostic file processor by the data collection function and produces a variety of reports and bar charts. The SRG is the equivalent of TPF “data reduction”.

STATMON. See NetView resource.

storage block. An area of storage that ALCS allocates to an entry. It is part of entry storage. See storage block sizes.

storage block size. ALCS allows storage blocks of up to 9 different sizes. These are identified in programs by the assembler symbols (or defined C values) L0, L1, L2, ..., L8. Installations need not define all these block sizes but usually define at least the following:

Size L0 contains 127 bytes of user data
Size L1 contains 381 bytes of user data
Size L2 contains 1055 bytes of user data
Size L3 contains 4000 bytes of user data
Size L4 contains 4095 bytes of user data.

The system programmer can alter the size in bytes of L1 through L4, and can specify the remaining block sizes.

storage level. An area in the ECB or a DECB used to hold the address and size of a storage block. See ECB level and DECB level.

storage unit. The ALCS storage manager allocates storage in units called storage units. Entry storage is suballocated within storage units; for example, one storage unit can contain an ECB and several storage blocks attached to that ECB.

ALCS uses three types of storage unit:

- Prime and overflow storage units for entry storage (also called type 1 storage units).
- High-level language storage units for stack storage (also called type 2 storage units).

- Storage units for heap storage for programs (also called type 3 storage units).

The size of a storage unit, and the number of each type of storage unit, is defined in the ALCS generation. See entry storage.

store access. Access which only involves writing (not reading). Compare with fetch access.

striping. A file organization in which logically adjacent records are stored on different physical devices. This organization helps to spread accesses across a set of physical devices.

Structured Query Language (SQL). a standardized language for defining and manipulating data in a relational database.

symbolic line number (SLN). In TPF, a 1-byte address of an ALC link, derived from the line number but adjusted so that all ALC links connected to the TPF system have a different symbolic line number. See also line number.

Synchronous Data Link Control (SDLC). A discipline conforming to subsets of the Advanced Data Communication Control Procedures (ADCCP) of the American National Standards Institute (ANSI) and High-level Data Link Control (HDLC) of the International Organization for Standardization, for managing synchronous, code-transparent, serial-by-bit information transfer over a link connection.

Transmission exchanges can be duplex or half-duplex over switched or nonswitched links. The configuration of the link connection can be point-to-point, multipoint, or loop.

Synchronous Link Control (SLC). A discipline conforming to the ATA/IATA Synchronous Link Control, as described in the ATA/IATA publication *ATA/IATA Interline Communications Manual*, ATA/IATA document DOC.GEN 1840.

syncpoint. An intermediate or end point during processing of a transaction at which the transaction's protected resources are consistent. At a syncpoint, changes to the resources can safely be committed, or they can be backed out to the previous syncpoint.

system error. Error that the ALCS monitor detects. Typically, ALCS takes a dump, called a system error dump, to the ALCS diagnostic file. See also ALCS diagnostic file and ALCS diagnostic file processor. See also system error dump, system error message.

system error dump. (1) A storage dump that ALCS writes to the ALCS diagnostic file when a system error occurs. See also ALCS diagnostic file and system error. (2) The formatted listing of a storage dump produced by the ALCS diagnostic file processor. See also ALCS diagnostic file processor.

system error message. A message that ALCS sends to receive only CRAS when a system error occurs. See also receive only CRAS and system error.

system error option. A parameter that controls what action ALCS takes when it detects a system error. See also system error.

system fixed file. An ALCS record class — one of the classes that reside on the real-time database. All system fixed-file records are also allocatable pool records (they have a special status of “in use for system fixed file”).

System fixed-file records are reserved for use by ALCS iteslf. See real-time database, record class, and record type.

system macro trace block. There is one system macro trace block. Each time an entry issues a monitor-request macro (or equivalent C function), ALCS records information in the system macro trace block.

This information includes the ECB address, the macro request code, the name of the program that issued the macro, and the displacement in the program. The ALCS diagnostic file processor formats and prints the system macro trace block in ALCS system error dumps. See also entry macro trace block.

System Modification Program/Extended (SMP/E). An IBM licensed program used to install software and software changes on MVS systems. In addition to providing the services of SMP, SMP/E consolidates installation data, allows flexibility in selecting changes to be installed, provides a dialog interface, and supports dynamic allocation of data sets.

Systems Application Architecture* (SAA*). A set of software interfaces, conventions, and protocols that provide a framework for designing and developing applications with cross-system consistency.

Systems Network Architecture (SNA*). The description of the logical structure, formats, protocols, and operational sequences for transmitting information units through, and controlling the configuration and operation of networks.

system sequential file. A class of sequential data sets used by ALCS itself. Includes the ALCS diagnostic file, the ALCS data collection file, and the ALCS update log file or files.

system state. The ALCS system can run in any of the following system states: IDLE, CRAS, message switching (MESW), and normal (NORM).

Each state represents a different level of availability of application functions. Altering the system state is called "cycling the system". See also standby.

system test compiler (STC). An offline ALCS utility that compiles data onto data files for loading on to the real-time database. STC also builds test unit tapes (TUTs) for use by the system test vehicle (STV).

system test vehicle (STV). An online ALCS function that reads input messages from a general sequential file test unit tape (TUT) and simulates terminal input. STV intercepts responses to simulated terminals and writes them to the ALCS diagnostic file.

T

terminal. A device capable of sending or receiving information, or both. In ALCS this can be a display terminal, a printer terminal, or a NetView operator identifier.

terminal address (TA). In ALC, the 1-byte address of an ALC terminal. Different terminals connected to the same terminal interchange have different terminal addresses. Different terminals connected to different terminal interchanges can have the same terminal address. See also terminal interchange.

terminal circuit identity (TCID). Synonym for line number.

terminal hold. When an ALCS application receives an input message, it can set terminal hold on for the input terminal. Terminal hold remains on until the application sets it off. The application can reject input from a terminal that has terminal hold set on. Also referred to as AAA hold.

terminal interchange (TI). In ALC, synonym for terminal control unit.

terminate. (1) To stop the operation of a system or device. (2) To stop execution of a program.

test unit tape (TUT). A general sequential file that contains messages for input to the system test vehicle (STV). TUTs are created by the system test compiler (STC).

time available supervisor (TAS). An ALCS or TPF function that creates and dispatches low priority entries.

time-initiated function. A function initiated after a specific time interval, or at a specific time. In ALCS this is accomplished by using the `CRETC` monitor-request macro or equivalent C function. See create service.

TP profile. The information required to establish the environment for, and attach, an APPC/MVS transaction program on MVS, in response to an inbound allocate request for the transaction program.

trace facility. See ALCS trace facility, generalized trace facility, and SLC link trace.

transaction. The entirety of a basic activity in an application. A simple transaction can require a single input and output message pair. A more complex transaction (such as making a passenger reservation) requires a series of input and output messages.

Transaction Processing Facility (TPF). An IBM licensed program with many similarities to ALCS. It runs native on IBM System/370 machines, without any intervening software (such as MVS). TPF supports only applications that conform to the TPF interface. In this book, TPF means Airline Control Program (ACP), as well as all versions of TPF.

Transaction Processing Facility Database Facility (TPFDF). An IBM licensed program that provides database management facilities for programs that run in an ALCS or TPF environment.

Transaction Processing Facility/Advanced Program to Program Communications (TPF/APPC). This enables LU 6.2 for TPF.

Transaction Processing Facility/Data Base Reorganization (TPF/DBR). A program which reorganizes the TPF real-time database.

Transaction Processing Facility/MVS (TPF/MVS). Alternative name for ALCS V2.

Transaction program identifier (TP_ID). A unique 8-character token that APPC/MVS assigns to each instance of a transaction program. When multiple instances of a transaction program are running sumultaneously, they have the same transaction program name, but each has a unique TP_ID.

transaction scheduler name. The name of an APPC/MVS scheduler program. The ALCS transaction scheduler name is ALCSx000, where x is the ALCS system identifier as defined during ALCS generation.

transfer vector. An ALCS application program written in assembler, SabreTalk, or C, can have multiple entry points for dynamic program linkage. These entry points are called transfer vectors. Each transfer vector has a separate program name.

transmission status indicator. See SLC transmission status indicator.

transmission sequence number. See SLC transmission sequence number.

trigger event. In message queuing, an event (such as a message arriving on a queue) that causes a queue manager to create a trigger message on an initiation queue.

trigger message. In message queuing, a message that contains information about the program that a trigger monitor is to start.

trigger monitor. In message queuing, a continuously-running application that serves one or more initiation queues. When a trigger message arrives on an initiation queue, the trigger monitor retrieves the message. When ALCS acts as a trigger monitor, it uses the information in the trigger message to start an ALCS application that serves the queue on which a trigger event occurred.

triggering. In message queuing, a facility that allows a queue manager to start an application automatically when predetermined conditions are met.

TSI exhaustion. The condition of an SLC channel when a sender cannot transmit another SLC link data block (LDB) because the maximum number of unacknowledged LDBs has been reached. The sender must wait for acknowledgement of at least one LDB so that it can transmit further LDBs. See also SLC channel, SLC link data block, SLC transmission sequence number, and SLC transmission status indicator.

two-phase commit. A protocol for the coordination of changes to recoverable resources when more than one resource manager is used by a single transaction. Contrast with single-phase commit.

type. See record type.

Type A traffic. ATA/IATA conversational traffic – that is, high-priority low-integrity traffic transmitted across an SLC or AX.25 link.

Type B application-to-application program (BATAP). In any system (such as ALCS) that communicates with SITA using AX.25 or MATIP, this is the program which receives and transmits type B messages.

Type B traffic. ATA/IATA conventional traffic – that is, high-integrity, low-priority traffic transmitted across an SLC or AX.25 link or a MATIP TCP/IP connection.

type 1 pool file dispense mechanism. The mechanism used in ALCS prior to V2 Release 1.3 (and still available in subsequent releases) to dispense both short-term and long-term pool-file records.

type 1 storage unit. Prime or overflow storage unit for entry storage. See storage unit.

type 2 pool file dispense mechanisms. The mechanisms available since ALCS V2 Release 1.3 to dispense pool-file records (the mechanisms are different for short-term and long-term pool-file records).

IBM recommends users to migrate to type 2 dispense mechanisms as part of their migration process.

type 2 storage unit. High-level language storage unit for stack storage. See storage unit.

type 3 storage unit. Storage unit for heap storage for programs. See storage unit.

U

unit of recovery. A recoverable sequence of operations within a single resource manager (such as WebSphere MQ for z/OS or DB2 for z/OS). Compare with unit of work.

unit of work. A recoverable sequence of operations performed by an application between two points of consistency. Compare with unit of recovery.

Universal Communications Test Facility (UCTF). An application used by SITA for SLC protocol acceptance testing.

update log. See ALCS update log.

user data-collection area. An optional extension to the data-collection area in the ECB. Application programs can use the DCLAC macro to update or read the user data-collection area.

user exit. A point in an IBM-supplied program at which a user exit routine can be given control.

user exit routine. A user-written routine that receives control at predefined user exit points. User exit routines can be written in assembler or a high-level language.

V

version number. In ALCS and TPF, two characters (not necessarily numeric), optionally used to distinguish between different versions of a program. Sometimes also used with other application components such as macro definitions.

virtual file access (VFA). An ALCS caching facility for reducing DASD I/O. Records are read into a buffer, and subsequent reads of the same record are satisfied from the buffer. Output records are written to the buffer, either to be written to DASD – immediately or at a later time – or to be discarded when they are no longer useful.

virtual SLC link. Used to address an X.25 PVC or TCP/IP resource for transmitting and receiving Type B traffic. Some applications (such as IPARS MESW) address communication resources using a symbolic line number (SLN) instead of a CRI. These applications can address X.25 PVC and TCP/IP resources by converting the unique SLN of a virtual SLC link to the CRI of its associated X.25 PVC or TCP/IP resource.

W

WAS Bridge. The ALCS WAS Bridge allows ALCS application programs to send and receive messages using optimized local adapters (OLA) for WebSphere Application Server for z/OS without the need to code those callable services in ALCS programs. The ALCS WAS Bridge installation-wide monitor exits USRWAS3, USRWAS4, USRWAS5, and USRWAS6 allow you to customize the behaviour of the WAS Bridge to suit your applications.

WebSphere* MQ for z/OS. An IBM product that provides message queuing services to systems such as CICS, IMS, ALCS or TSO. Applications request queuing services through MQI.

wide character. A character whose range of values can represent distinct codes for all members of the largest extended character set specified among the supporting locales. For the z/OS XL C/C++ compiler, the character set is DBCS, and the value is 2 bytes.

workstation trace. One mode of operation of the ALCS trace facility. Workstation trace controls the remote debugger facility. The remote debugger is a source level debugger for C/C++ application programs.

World Trade Teletypewriter (WTTY). Start-stop telegraph terminals that ALCS supports through Network Terminal Option (NTO).

Z

Z message. See ALCS command.

Bibliography

ALCS Version 2 Release 4.1 publications

- *Application Programming Guide*, SH19-6948
- *Application Programming Reference – Assembler Language*, SH19-6949
- *Application Programming Reference – C Language*, SH19-6950
- *Concepts and Facilities*, SH19-6953
- *General Information Manual*, GH19-6738
- *Installation and Customization*, SH19-6954
- *Licensed Program Specifications*, GH19-6805
- *Messages and Codes*, SH19-6742
- *Operation and Maintenance*, SH19-6955

Other publications

- *An Introduction to Messaging and Queueing*, GC33-0805
- *BookManager Read and Build General Information*, GC23-0447
- *DB2 UDB Server for z/OS and OS/390 What's New?*, GC26-9946
- *High Level Assembler for MVS & VM & VSE General Information*, GC26-4943
- *ISPF and ISPF/PDF General Information*, GC34-4250
- *Language Environment Concepts Guide*, SA22-7567
- *MVS Data Facility Sort General Information*, GC33-4033
- *SMP/E User's Guide*, SA22-7773
- *TPF V4R1 General Information*, GH31-0147
- *TPFDF R1 General Information*, GH31-0177
- *z/OS DFSMS Introduction*, SC26-7397
- *z/OS Introduction and Release Guide*, GA22-7502

Readers' Comments — We'd Like to Hear from You

Airline Control System Version 2
General Information Manual
Release 4.1

Publication No. GH19-6738-09

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	□	□	□	□	□

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	□	□	□	□	□
Complete	□	□	□	□	□
Easy to find	□	□	□	□	□
Easy to understand	□	□	□	□	□
Well organized	□	□	□	□	□
Applicable to your tasks	□	□	□	□	□

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? □ Yes □ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
GH19-6738-09

Cut or Fold
Along Line

Fold and Tape | **Please do not staple** | Fold and Tape

PLACE
POSTAGE
STAMP
HERE

ALCS Development
2455 South Road
P923
Poughkeepsie NY 12601-5400
USA

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

Program Number: 5695-068

GH19-6738-09