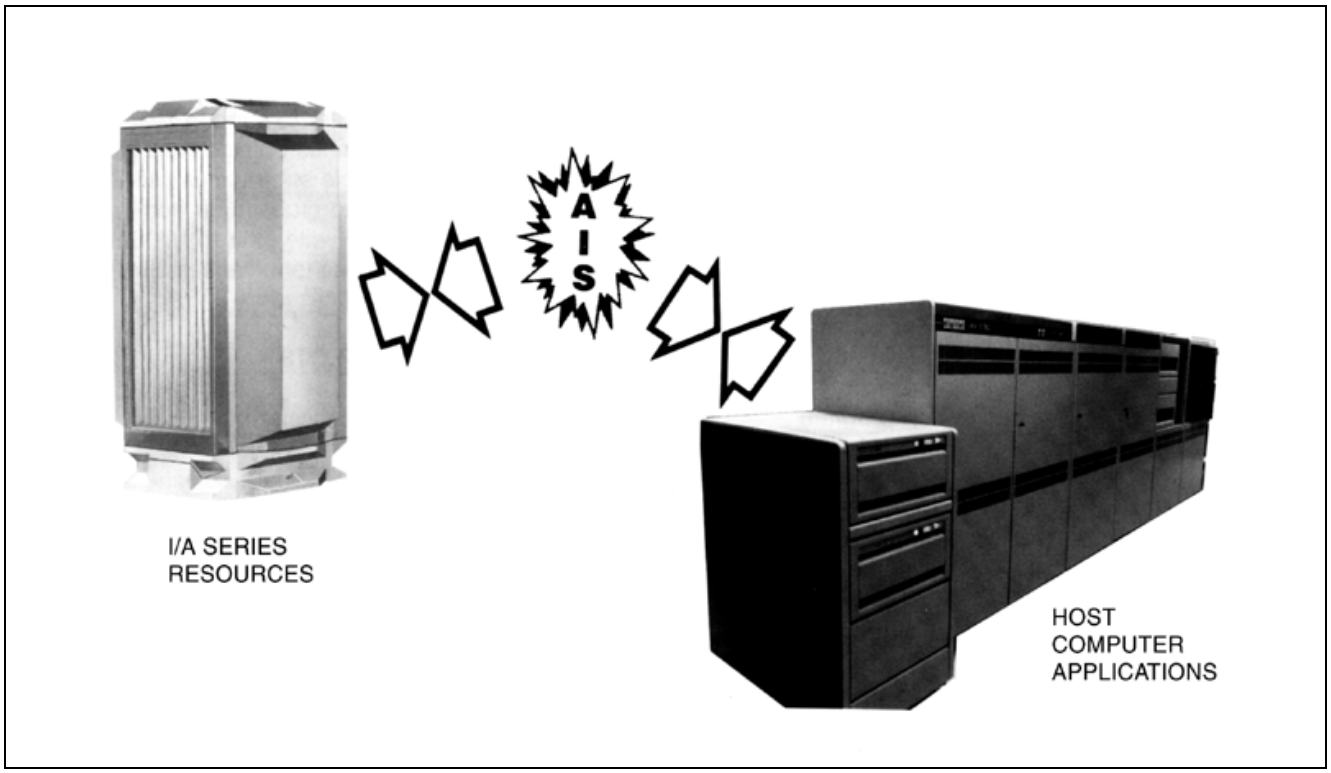
FOXBORO®

PSS 21S-4N3 B3

I/A Series® Software
Application Interface Software
for Digital Equipment Corporation VAX Computers



Application Interface Software (AIS) provides an application program interface (API) resident in a host computer. Host computer applications use AIS to gain access to objects, files, and/or virtual terminal interface in connected I/A Series systems. This application program interface, following international standards, insulates the customer application programs from host computer specifics. Customer applications deal with functional requirements instead of communication implementation details.

Host computer programmers write application programs using AIS subroutines to perform such functions as process variable access, file transfer, and terminal interface. The AIS routines work in conjunction with one or more Information Network Interface (INI) modules configured in connected I/A Series systems. AIS works with Information Network Interface modules direct or network connected to the host computer system. The Host computer vendor determines appropriate Host system hardware and software to support the connection.

Application functions available with AIS:

- Non-connected named process variable access
- Connected named process variable access
- Exception based, named process variable access
- Named file upload and download
- User callable floating point data conversion routine
- Named file partial upload
- Programmatic virtual terminal interface
- Redirection of virtual terminal input and output to and from files
- Menu-driven virtual terminal access
- Menu-driven file transfer

Invensys
An Invensys company

Product Specifications

AIS optionally provides circuit optimization and data object path redundancy on multiple INI systems. The AIS package, by configuration, apportions process variable lists among several Information Network Interface modules. AIS also uses configurable alternate INI paths for object access upon failure of a particular INI. This maintains operating object access services originally provided by the failed INI path.

AIS additionally offers application program development tools. This package includes utility programs for exercising the package functions and for comparing results during application development. Several optimizations are configurable for tuning the performance of the host computer application under development.

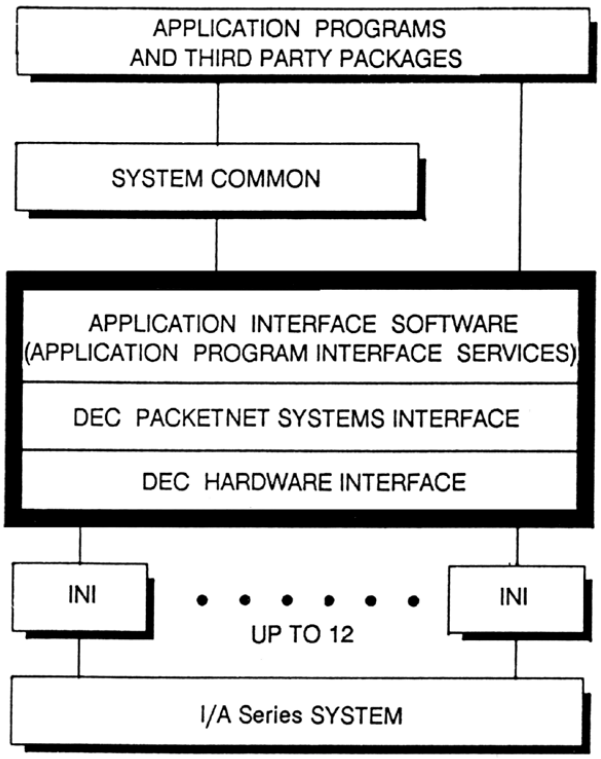


Figure 1. Layered Application Program Interface

AIS SERVICES

The Application Interface Software (AIS) accesses data objects, files residing in the I/A Series stations, and Application Processor terminal sessions for user-written application programs resident in Host computers. Figure 2 illustrates the relationship between Host computer programs and I/A Series services.

AIS runs in a Host computer to support redundant program access of I/A Series objects via INIs. It includes FORTRAN-callable subroutines used by the Host programs to read global data from I/A Series Processors, to write data to I/A Series Processors, and to execute terminal sessions in I/A Series Application Processors.

ACCESSING DATA OBJECTS

The methods of object access are:

- Unbuffered read/write of data objects.
- Buffered read/write of data objects.
- Buffered read/write with continuous updates.
- Buffered read/write with continuous updates and queueing of all changes.

Unbuffered Read/Write of Data Objects

This access method is appropriate when the data objects are accessed infrequently. Data objects of all value types can be read or written. Both connectable and non-connectable data objects may be accessed. Related subroutines are:

- uread
- uwrite
- sread
- swrite

UNBUFFERED READ OF NON-STRING DATA OBJECTS

The caller of the uread subroutine specifies the gateway paths to be used and a set of data objects by name. The subroutine returns the value, status, and error code for each non-string data object.

UNBUFFERED WRITE OF NON-STRING DATA OBJECTS

The caller of the uwrite subroutine specifies the gateway paths to be used, a set of data objects by name, and the new values for the data objects. All data objects that are accessible are updated. An error code is returned for each data object.

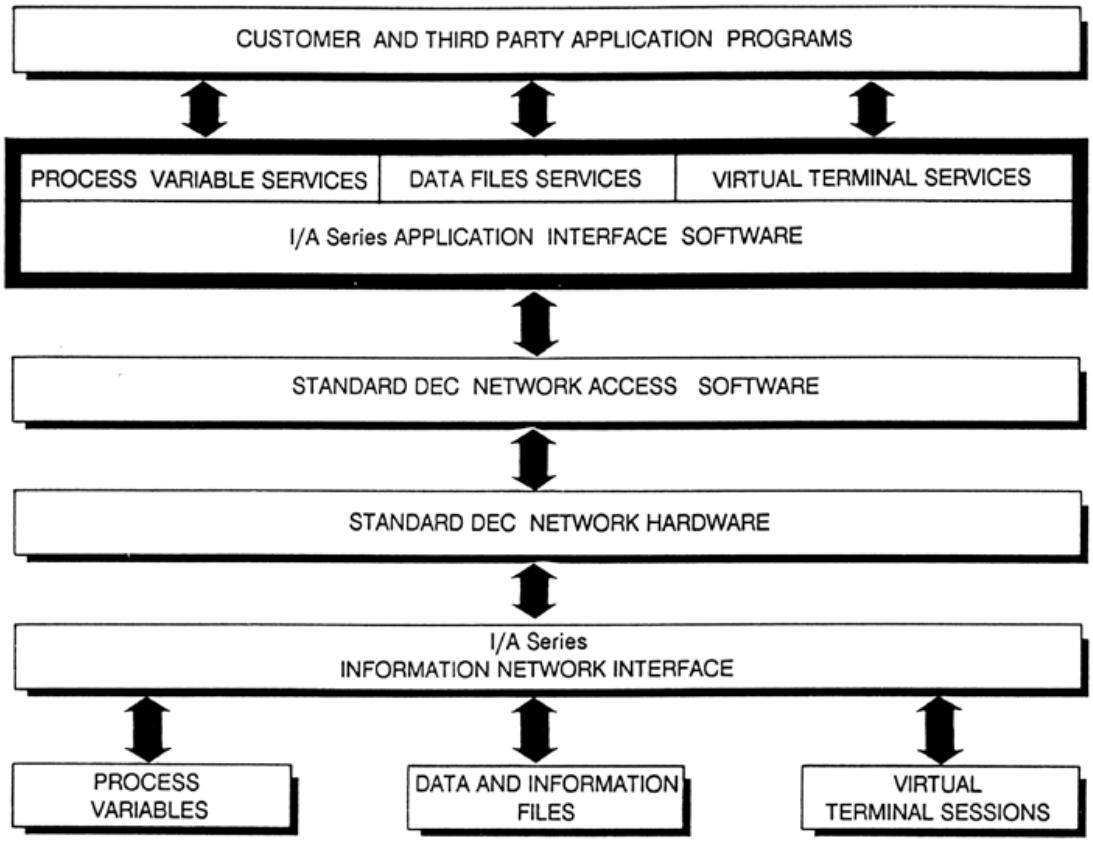


Figure 2. Host Services Relationship

UNBUFFERED READ OF STRING DATA OBJECTS

One data object may be read at a time. The caller of the sread subroutine specifies the gateway path to be used and the name of a data object. The subroutine returns the string, the status, and an error code for the data object.

UNBUFFERED WRITE OF STRING DATA OBJECTS

One string data object may be written at a time. The caller of the swrite subroutine specifies the gateway path to be used, the name of a data object, and the string. The subroutine returns the status and an error code for the data object.

Buffered Read/Write of Data Objects

This access method is appropriate when data objects are being accessed frequently. Only connectable data objects can be accessed. Related subroutines are:

- bopen
- sbopen
- bread
- bwrite
- clsset

SPECIFICATION OF DATA OBJECTS FOR BUFFERED READ/WRITE

The bopen subroutine establishes the change-driven connections for a set of data objects. The data objects are then accessible by using the bread and bwrite subroutines. The caller optionally specifies the gateway paths. The caller specifies a set of data objects by name, change value (delta), assumed value type, an indicator if the set is to be closed when the program exits, and an access mode (read-only or read/write). The subroutine makes the connections to the data objects and returns a data set number to be used in other calls to reference the set of data objects, a status, and an error code per data object informing the caller if and why a connection cannot be made.

BUFFERED READ OF DATA OBJECTS

The caller of the bread subroutine specifies a set of data objects by data set number. The subroutine returns the value and status for each data object in the data set.

BUFFERED WRITE OF DATA OBJECTS

The caller of the bwrite subroutine specifies a set of data objects by data set number and the new values. All data objects that are accessible are written. An error code is returned for each data object in the data set.

Buffered Read/Write with Continuous Update

This access method is appropriate when you want to have values updated automatically in the Host without doing any bread calls. Related calls are:

- copen
- scopen
- bread
- bwrite
- clsset

The copen call establishes the change-driven connections and starts the updating process. The caller optionally specifies the gateway path(s) to be used. The caller specifies a set of data objects by name, change value (delta), assumed value type, an indicator if the set is to be closed when the program exits, access mode (read-only or read/write), and an array to return the indexes into system common arrays where the value and status are stored. The subroutine returns a data set number to be used in the bread, bwrite, and clsset calls.

In using the scopen, it is also possible to specify the frequency of checking the delta. By updating the write value table in system common with wrtval, it is possible to invoke a change-driven write of the value to the I/A Series object.

Note that bread calls are unnecessary, because value and status information is automatically updated to system common arrays in the Host; however, breads are possible.

The readval, readwval, readsta, readccnt, readwcnt, mreaidx, and mreawidx calls return read and write value, status, read and write change count and multiple read and write information from the AIS value, status and change count shared memory arrays.

Buffered Read/Write with Continuous Update and Queueing of Changes

This access method is appropriate when you want to be made aware of every change in value greater than or equal to the delta and/or every change in status of object in the set.

Related subroutines are:

- qopen
- sqopen
- qread
- bread
- bwrite
- clsset

The qopen call establishes the change-driven connections and starts the updating and queueing process. The caller optionally specifies the gateway path(s) to be used. The caller specifies a set of data objects by name, change value (delta), assumed value type, an indicator if the set is to be closed when the program exits, and an array to return the indexes into the system common array where the value and status are stored. The subroutine returns a data set number to be used in the qread, bread, bwrite, and clsset calls.

In using the sqopen, it is also possible to specify the frequency of checking the delta. By updating the write value table in system common with wrtval, it is possible to invoke a change-driven write of the value to the I/A Series object.

The readval, readwval, readsta, readccnt, readwcnt, mreaidx, and mreawidx calls return read and write value, status, read and write change count and multiple read and write information from the AIS value, status and change count shared memory arrays.

EXTRACTING CHANGES FROM A CHANGE QUEUE

The caller of the qread subroutine specifies a set of data objects by data set number. The subroutine returns the changes (value and status) from its queue for that data set. Each change is accompanied by a data set entry number. Some data objects may have no changes associated with them; others may have several changes. The maximum number of changes to be returned is specified by the caller. The actual number of changes is returned to the caller. A return code indicates the state of the queue.

CLOSING A SET OF DATA OBJECTS

Data sets previously opened with the bopen, copen, qopen, sbopen, scopen, or sqopen call can be closed with the clsset call.

Obtaining Information about a Data Set

AIS provides routines that can optionally be used to aid in developing an application. These routines return information that is available when the data set is opened. Providing this information relieves the application from housekeeping. Related subroutines are gsinfo, getnam, getscn, gsnent, get_set_name, and get_set_num.

Converting Floating Point Values

For data object access calls, AIS converts floating point data between Host and I/A Series formats. For file access calls, AIS performs no floating point conversion of data values, but provides a routine for that purpose. Related subroutine is cnvrt.

Optimizing the Use of Virtual Circuits

By default, AIS subroutines which communicate to INIs establish and release the X.25 virtual circuit to the INIs each time they are called. This approach allows more applications to make use of the INI services.

If you have an application which makes frequent calls to such subroutines, you can optimize the application by specifying that virtual circuits are to be established once, kept for as long as needed, then released. Related subroutines are getcir and relcir.

Redundancy

The goal of redundancy is to keep as many objects accessible as possible when INI(s) fail.

To supply redundancy it is necessary to have two or more INIs in the same group with some free lists. The redundancy algorithms keep as many lists open on the preferred INI as possible, or, as a second choice within a group. The result is that lists are moved from a failing preferred INI to a functional INI, then back to their preferred INI as soon as the preferred INI returns to a functional state (“OK”).

The state of the INI is constantly monitored by the AIS package. If one of the INIs changes state (from OK to FAILED or FAILED to OK), then the following actions are performed.

- All lists not currently opened on any INI are opened on their preferred INI. AIS opens these lists if the INI is OK and has available resources. Unopened lists occur when other INIs have no available resources to back up a failed INI.
- All lists currently opened on backup INIs are moved to their preferred INI. AIS opens these lists if the preferred INI is OK and has available resources.
- All lists not currently opened are opened by AIS on any INI in the same group. AIS opens these lists if the target INI is OK and has available resources.

ACCESSING I/A Series FILES

There are three methods of file access:

- Transferring a binary file between I/A Series Application Processor and the Host.
- Transferring an ASCII file between I/A Series Application Processor and the Host.
- Reading part of a file from I/A Series Application Processors.

The file access allows the reading and writing of binary and ASCII files. The accessing can be done through FORTRAN interface calls or by using a utility program, FILUTL.

The iarfil routine transfers the entire contents of an I/A Series file to a binary Host file.

The iawfil routine transfers the entire contents of a binary Host file to an I/A Series file.

The iartxt routine transfers the entire contents of an I/A Series file to a text Host file.

The iawtxt routine transfers the entire contents of a text Host file to an I/A Series file.

The pfread routine transfers a specified portion of an I/A Series file into the memory space of the application program in the Host.

Reading a File from I/A Series Application Processors

The iarfil, I/A Series read file subroutine, copies a binary file from an I/A Series Application Processor to a Host. The caller specifies the I/A Series file name, Host file name, gateway path, user id, group id, and Application Processor logical name (as configured in the INI). An error code is returned. The file content is copied verbatim with no consideration of file type. Floating point data is not converted from the I/A Series floating point format to the Host floating point format.

The iartxt routine is identical to iarfil except that it also converts the file contents to the proper text format on the Host.

Writing a File to I/A Series Application Processors

The iawfil, I/A Series write file subroutine, copies a binary file from a Host to an I/A Series Application Processor. The caller specifies the I/A Series file name, Host file name, gateway path, user id, group id, and Application Processor logical name (as configured in the INI). An error code is returned. The file content is copied verbatim with no consideration of file type. Floating point data is not converted from the Host floating point format to the I/A Series floating point format.

The Iawtxt routine is identical to iawfil except that it also converts the file contents to the proper text format on the I/A Series station.

Reading Part of a File from I/A Series Application Processors

The pfread, I/A Series partial read file subroutine, copies part of a file from an I/A Series application processor into the caller's program space. The caller specifies the I/A Series file name, gateway path, userid, group id, logical name, the number of bytes to read, and the array in which to store the information. An error code is returned. The file content is copied verbatim with no consideration of file type. Floating point data is not converted from the I/A Series floating point format to the Host floating point format.

ACCESSING VIRTUAL TERMINAL SESSIONS

The virtual terminal access allows the use of UNIX features from a Host terminal and a Host program. The accessing can be done by using a utility program, called VTUTIL, or through FORTRAN interface calls.

Establishing a Session

The vtinit subroutine establishes a session by logging into an I/A Series Application Processor. The caller specifies the gateway path, the Letterbug of the chosen I/A Series Application Processor, the login name and password of the account, and the primary prompt string of the UNIX operating system shell. An error code is returned.

Processing UNIX Command Lines

The vtprog subroutine issues UNIX command lines from a specified array and returns UNIX responses to a specified array. The caller specifies the array containing the UNIX command lines, the number of characters to read from the command line array, and the array in which to store the UNIX responses. The number of characters stored in the response array, and an error code are returned.

The vtcont subroutine has the same arguments as vtprog, but its function is to "continue" the processing of UNIX command lines initiated with vtprog. A return code indicates when "continued" processing is necessary to complete the command(s).

Terminating a Session

The vtend subroutine terminates a session with an I/A Series Application Processor. An error code is returned.

UTILITIES

FILUTL Utility Program

The utility program, FILUTL, allows the user to copy:

- A binary Host file to an I/A Series file.
- An I/A Series file to a binary Host file.
- A part of an I/A Series file to VAX memory.
- A Host text file to an I/A Series file.
- An I/A Series text file to a Host, edt, file.

An I/A Series data format to Host floating point conversion routine is provided. Related subroutine is cnvrt.

VTUTIL Utility Program

The utility program, VTUTIL, allows the user to:

- enter UNIX command lines from a Host terminal keyboard and receive prompts/responses on the Host terminal screen.
- enter UNIX command lines from a file and optionally receive UNIX responses back in a file and/or to the Host terminal screen.
- enter UNIX command lines from a Host terminal keyboard and optionally receive UNIX responses back in a file and/or to the Host terminal screen.

The FORTRAN interface calls allow a Host program to issue UNIX command lines and receive responses back in the Host program. The interface calls are:

- the vtinit subroutine establishes a session by logging into an I/A Series Application Processor.
- the vtprog subroutine interacts with UNIX by issuing UNIX command lines from a specified array and returning UNIX responses to a specified array.
- the vtcont subroutine continues the processing of UNIX command lines initiated with vtprog.
- the vtend subroutine terminates a session with an I/A Series Application Processor.

AISTST Utility Program

AISTST exercises and inspects the AIS package. The following is provided:

- The capability to exercise all AIS subroutines except file access and virtual terminal access;
- The capability to report on the status of the package;
- The capability to turn traces on or off.

GWSTAT Utility Program

The gwstat utility program retrieves the status of INI modules and allows the user to connect or disconnect a gateway.

AISDIS Utility Program

The aisdis program initially displays the status of configured INI modules and the total number of change messages per INI module. Secondary displays give the name, value, and status for the data objects of a specified data set. Aisdis also displays the number of changes per object either read from or written to a connected I/A Series system.

Aisdis is useful for tuning change and write deltas for data objects. Aisdis is also used to monitor change and write message load on INI modules.

SPECIFICATIONS

This software release provides support for the Digital Equipment Corporation (DEC) VAX and microVAX computer systems running VMS Operating System Release 4.7 and later. AIS is a product which layers over the DEC Packetnet Systems Interface (PSI) Communication Package Release 4.2 or later for directly connected INI modules to DEC host computers. AIS also supports the Packetnet Systems Interface Access (PSI Access) communication package for INI modules connected to DECNet using DEC X25 router 2000 interface modules.

Host Hardware
Digital Equipment Corporation (DEC) VAX and MicroVAX Computers.

Bulk Storage Required
4350 Blocks (not including log files) plus 10 Blocks per INI serviced.

Direct Connection
Appropriate synchronous host computer hardware interface supporting CCITT X.25 communications as defined by Digital Equipment Corporation.

DECNet Connection
X25router 2000 MicroServer (DEMSA) and supporting hardware as defined by Digital Equipment Corporation.

Host Software
Digital Equipment Corporation (DEC) VMS Operating System V4.7 or later.

Direct Connection
Digital Equipment Corporation Packetnet Systems Interface (PSI) V4.2 or later.

DECNet Connection
X25portal 2000 license for each MicroServer and appropriate DEC Packetnet Systems Interface Access (PSI Access) software for each DEC host requiring access to I/A Series processor data.

I/A SERIES MODULES

Select from one to twelve Information Network Interface (INI) modules.

APPLICATION INTERFACE SOFTWARE (AIS)

VAX AIS-4 Software License works in conjunction with one to four Information Network Interface (INI) modules configured in connected I/A Series system(s).
VAX AIS-8 Software License works in conjunction with one to eight Information Network Interface (INI) modules configured in connected I/A Series system(s).
VAX AIS-12 Software License works in conjunction with one to twelve Information Network Interface (INI) modules configured in connected I/A Series system(s).

The Foxboro Company
33 Commercial Street
Foxboro, Massachusetts 02035-2099
United States of America
http://www.foxboro.com
Inside U.S.: 1-508-543-8750 or 1-888-FOXBORO (1-888-369-2676)
Outside U.S.: Contact your local Foxboro Representative.

MB 021 Printed in U.S.A. 1293