JES2 Spool Data Set Browse

Table of Contents

Preface

This document describes the application programming necessary to utilize the JES2 spool data set browse (SDSB) interface. Through the use of SDSB, an authorized program can dynamically allocate spool data sets. Once the data sets are allocated, standard I/O macros can be used to read the data.

It is assumed that the reader is familiar with dynamic allocation and the data management macros required for I/O.

Description

This document describes the interface used by an application program to invoke the function provided by JES2 spool data set browse (SDSB). Spool data set browse allows an application to dynamically allocate a spool data set and use standard I/O macros to read the file. The interface provides unwritten spool buffer support so that blocks not yet written to spool can be retrieved.

Allocation of spool data sets can only be performed by authorized programs. The data set being requested is specified in a data set token, which is passed to dynamic allocation along with the data set name.

Once the data set is allocated, the data set can be read using one of two methods:

1. Through the compatibilty interface (DCB, GET).
2. Through the ACB/RPL interface.

The compatibility interface can be used when synchronous sequential access is required.

The ACB/RPL interface can be used when random access to the spool file is required, or when asynchronous processing is to be done.

Allocation

JES recognizes a spool data set browse allocation when a data set token is specified on a dynamic allocation call. The data set token is specified in addition to the data set name and contains control information required by JES to allocate the desired data set.

Spool data set browse can only be specified on dynamic allocation requests.

The following text units are required on the dynamic allocation request:

- DALDSNAM (data set name, set from PDBDSNAM)
- DALSTATS (disposition = SHR)
- DALSSREQ (subsystem name)
- DALBRTKN (browse token)

Other text units are optional, such as DALRTDDN, to return the ddname allocated to the data set.

Note that the DALSSREQ text unit can only be specified by authorized programs, and hence the allocation must be done in authorized state. However, only the allocation requires authorization; after the allocation is complete, an unauthorized task can perform the I/O.

Prior to issuing the dynamic allocation request, the application must build the browse token and pass it with the DALBRTKN text unit. The format of the browse token is mapped by macro **IAZBTOKP**. The browse token is built in text unit format with 7 subparameters. As a result, when building the browse token, the following SVC 99 text unit fields must be set:

Field	**Value**
S99TUKEY	DALBRTKN
S99TUNUM	7
S99TUPAR	Mapped by IAZBTOKP

The token is fixed length and all subparameters must be coded. Each browse token subparameter contains a length followed by the data, so it will be in text unit format.

The fields in the token are completed as follows:

BTOKPL1 is the length of the browse token identifier (LENGTH(BTOKID)).

BTOKID is the browse token id (BTOK). The IAZBTOKP macro defines constant BTOKCID to be used to set this field.

BTOKPL2 is the length of the token version field (LENGTH(BTOKVER)).

BTOKVER is the version number of the token parameter list being used. The IAZBTOKP macro defines constant BTOKVRNM to be used to set this field.

BTOKPL3 is the length of the IOT MTTR field (LENGTH(BTOKIOTP)).

BTOKIOTP is the MTTR of the IOT containing the PDDB of the file to be allocated (obtained from JOEIOTTR or IOTTRACK, for example).

BTOKPL4 is the length of the job key field (LENGTH(BTOKJKEY)).

BTOKJKEY is the job key of the file to be allocated (obtained from JCTJBKEY, JQEJBKEY, or SJBJKEY, for example).

BTOKPL5 is the length of the asid field (LENGTH(BTOKASID)).

BTOKASID is the asid of the job if the job is active on the same system as the browse request is being made, or zero otherwise.

BTOKPL6 is the length of the RECVR field (LENGTH(BTOKRCID)).

BTOKRCID is the eight byte userid to be used as the RECVR on the SAF call used by JES2 to check authority to the browse request, or zeros if the RECVR is not being used. When RECVR is used, the value must be left justified and padded with blanks. When this parameter is specified, the logstr field should also be used so that usage of recvr can be logged. However, neither JES2 nor SAF enforce this convention.

BTOKPL7 is the length of the logstr field (LENGTH(BTOKLOGS)).

BTOKLSDL is the length of the logstr (specified in field BTOKLSDA) to be used on the SAF call used by JES2 to check authority to the browse request, or zero if the logstr is not being used.

The logstr length must be a value from 0 to 254.

BTOKLSDA is the text of the logstr if BTOKLSDL is non-zero, or zeros if the logstr is not being used. The maximum length text is 254 characters.

When the compatibility interface will be used to read the data set, text units specifying the record format, record length, and blocksize should be used.

Security

No SAF call will be done by JES2 during allocation.

At data set open time, JES2 will SAF check read access to the JESSPOOL resource associated with the data set. The resource name will be constructed using the data set name passed on the allocation request.

If the browse token specifies a recvr userid, the SAF call will be done with the RECVR parameter. When the recvr userid is specified, the logstr parameter should also be supplied.

If the data set fails the security check, the open request will be failed with R15=0C and an error code stored in ACBERFLG (decimal 152).

Errors and Return Codes

If an error occurs allocating the data set, dynamic allocation will return a reason and return code describing the error. The system dairfail service can be used to format the error text.

If an error occurs during data set open, a return code will be set in R15 following the OPEN and a reason code will be stored in the ACB.

Using the Compatibility Interface

After the spool data set has been allocated, the compatibility interface can be used to read each record sequentially.

The application builds and opens a DCB that specifies the ddname of the allocated spool data set. If the record format, record length, and block size were not specified on the allocation, they must be specified in the DCB.

The system will perform a SAF call as part of OPEN processing to ensure that the user is authorized to the data set. If the user is not authorized, a S913 ABEND will result.

After the DCB has been opened, a GET macro pointing to the DCB can be used to read the file.

When processing is complete, a CLOSE macro should be issued to close the file. The same task that opened the DCB must be used to close it.

Using the ACB/RPL Interface

An application can use the ACB/RPL interface to obtain the most flexibility when using spool data set browse.

With this method, the application builds and opens an ACB (access method control block), and uses RPL based macros to read and position the data set.

The ACB/RPL macros used are a subset of those documented for VSAM. In general, JES implements only those features required to process the data set. Other options specifed on the macros are ignored. When coding the ACB and RPL macros, AM=VSAM should be specified or defaulted.

Building the ACB

After the dynamic allocation for the data set is complete, the application must build and open an ACB. The ACB can be built using GENCB service, or can be mapped with the IFGACB macro. The storage for the ACB must be resident below the 16 megabyte line.

The SHOWCB and MODCB services can be used to display and modify selected fields of the ACB. However, some of the fields required by JES are not processed by those services, and hence they may be of limited use.

The only required field for the ACB is the ddname to use. The ddname can be assigned when the spool data set is allocated, or the system can return a generated name. In either case, the ACBDDNAM must be completed before the ACB is opened.

The ACB can also specify an optional exit list. The exit list is built with the EXLST macro, and can be used to specify the EODAD, SYNAD, and LERAD exits.

Once the ACB is generated, it is opened using an OPEN macro. As part of open, the system will perform a SAF call to ensure that the user is authorized to the data set. If the user is not authorized, a S913 ABEND will result. However, unless authorization has changed between the time the data set was allocated and opened, an unauthorized user will be detected at allocation time.

If the open is successful, RPL based macros can be used to read the data set.

If the open is not successful, error and reason codes will be placed in the ACB that describe the error.

When processing is complete, the application should issue a CLOSE macro specifying the open

ACB. However, if a CLOSE is not done, a close will be performed automatically by the system when end of task occurs. The close must be done by the same task that opened the ACB.

Requesting Carriage Control

The spool data set browse interface can be used to obtain the carriage control character if it is contained in the record.

The spool data set browse interface will return carriage control with the record only if it is requested.

The application indicates that carriage control is wanted by setting flag ACBCCTYP as follows:

- ACBCCTYP.ACBCCASA = ON indicates that ASA characters are wanted.
- ACBCCTYP.ACBCCMCH = ON indicates that machine characters are wanted.

Both bits can be turned on to indicate that carriage control is wanted regardless of format. When a data set contains carriage control, and the application requests carriage control for that type, the control character will be returned in the first byte of the record area returned by JES.

If the data set does not contain carriage control, but the application requests carriage through the setting of ACBCCTYP, JES will do the following:

- If ACBCCASA is on, JES will return a X'40' as the carriage control.
- If ACBCCMCH is on, JES will return a X'09' as the carriage control.
- If both ACBCCASA and ACBCCMCH are on, JES will return a X'09' as the carriage control.

Using RPL Based Macros

All I/O requests are specified using an RPL that contains the address of an open ACB. The RPL can be built using the GENCB service, or can be mapped with the IFGRPL macro. SHOWCB and MODCB can also be used for some functions. The storage for the RPL must be resident below the 16 megabyte line.

The processing options are specified in the RPLOPTCD parameter. Only a subset of those defined by VSAM are recognized by JES. In particular, spool data set browse supports only the move mode (OPTCD=MVE) of processing. OPTCD=SYN can be specified for synchronous requests (the default), or OPTCD=ASY can be used for asynchronous processing.

The application must obtain a buffer area to contain the record read by JES. The address of the area is placed in RPLAREA and its length in RPLBUFL. If the area is not large enough to contain the record, JES will set an error code in the RPL. The storage for the buffer may be resident above the 16 megabyte line.

Each record is read using a GET macro which points to the RPL. When synchronous processing is used, control will return to the application when the GET is complete and the record has been moved to the application provided area. The length of the returned record is set by JES in field RPLRLEN.

When asynchronous processing is used, the application must issue a CHECK macro specifying the RPL to be waited on. Control will return to the application when the function specified by the RPL is complete.

After each GET request, JES returns a token that can be used to position directly to the record if it needs to be reread. JES returns the 8 byte token in field RPLRBAR after each GET. The application should treat RPLRBAR strictly as a token, and not depend on it to be in a specific format.

The POINT macro can be used to locate a previously read record. POINT specifies an RPL that contains an RPLRBAR that was returned on the GET request for the desired record. POINT positions JES to the record to be read; it does not read the record. A GET macro must be issued after the POINT to retrieve the record.

Each GET request returns a logical record in the file. The application is isolated from the internal format of the record. JES places the complete record in the RPLAREA buffer and its length in the RPLRLEN field. JES performs the necessary unwritten buffer support for active jobs; however, no indication of the source of the record is placed in the RPL.

JES places feedback information in the RPL after each request. RPLRTNCD and RPLCNDCD should be checked for satisfactory completion after each operation.

End of File Processing

For jobs not active in execution, the spool data set browse interface will drive the end of file exit when no more data can be read.

When using the compatibility inteface, the exit is defined using the EODAD parameter of the DCB.

When using the ACB/RPL interface, the exit is defined on the EXLST macro. In addition, the return codes are placed in the RPL to indicate the end of file condition.

For jobs active in execution, the interface will attempt to read unwritten spool buffers that reside in the address space of the active job. When all unwritten buffers have been read, the interface will drive the end of file exit. However, the application can issue subsequent get requests to attempt to obtain additional data. On each get request, the interface will return the unwritten buffer data, if available.

Thus, an end of file condition for an active job should be considered temporary rather than permanent. As the active job creates additional data, subsequent get requests can be used to retrieve it. If no more data is available at the time of the get, end of file will be driven.

This processing differs from standard access methods. Normally, a get request issued after end of file is considered as a permanent error. However, this condition should be expected when using the spool data set browse interface against an active job.

Performance

Spool data set browse presents a record level interface, i.e., a complete record is returned on each get request. If a record is internally stored in several JES2 spool blocks, the interface will perform the necessary spool I/O to assemble the record segments. JES will maintain only one spool buffer in storage at a time.

Similarly, if a point macro is issued for a record not contained in the last spool block, JES will initiate the I/O to read the block.

Performance through the interface can be optimized by limiting I/O requests. Therefore, it may be desirable to buffer previously read records in storage, rather than call the interface with POINT and GET to reread a record.

Secondary Subsystem Support

The spool data set browse interface supports allocation of spool data sets to secondary JES2 subsystems. The subsystem name specified on the DALSSREQ text unit directs the allocation to the proper JES.

Note that a secondary subsystem cannot be halted until all outstanding allocations are freed. _