REXX/VSAM Interface

Written By: Steven W. Scott

Overview

The RXVSAM REXX interface to VSAM consists of three modules: an assembler stub which acts as a REXX function (RXVSAM), an assembler macro which facilitates access to the VSAM handler (VSAMIO), and an assembler program which is the actual VSAM handler (VSIOMOD). The VSAM specific modules were designed to function independent of the REXX stub program and can be used by any module capable of construction the parameter list.

Usage

The current implementation of the interface requires that a four byte variable, specifically named “VSANCHOR”, must be created before any calls to the RXVSAM function are made, and must be initialized with the character value “INIT”. If the variable does not exist the function returns an error code indicating so. This variable acts as the primary anchor address for the underlying VSAM structures and request blocks.

The load library containing VSIOMOD must be present in the STEPLIB concatenation for any job step using the interface, and RXVSAM must be present in either the ISPLLIB concatenation or STEPLIB. Calls made to RXVSAM follow the following format:

RC = RXVSAM(DDNAME, FUNCTION, PARAM1, PARAM2)

The parameters are described as follows:

DDNAME – This is the DDNAME assigned to the previously allocated VSAM dataset.

FUNCTION – This parameter defines the function to be performed against the VSAM dataset. Available functions are:

OPEN – This function requests that the dataset is to be opened for read only access.

OPENU – This function requests that the dataset is to be opened for update purposes.

OPENL – This function requests that the dataset is to be reset to empty status, and opened for data load purposes.
OPENRU - This function requests that the dataset be reset to empty status, then write one dummy record with a key of x'00's, and then close and re-open the dataset for updating.

CLOSE – This function requests that the dataset is to be closed.

READ – This function requests that a record is to be read into the data stack.

READU – This function requests that a record is to be read with the intent to update. If no update is performed an ENDREQ must be issued.

WRITE – This function requests that a record which has been placed on the data stack is to be written to the VSAM file.

DELETE – This function requests that the current record is to be deleted.

POINT – This function requests that a VSAM POINT is to be executed against the dataset.

ENDREQ – This function requests that an ENDREQ is to be performed against the dataset.

PARAM1, PARAM2 – These are optional parameters which are available according to the function requested. Available parameters by function are:

OPEN – No additional parameters available.

OPENU - No additional parameters available.

OPENL - No additional parameters available.

OPENRU - No additional parameters available.

CLOSE – No additional parameters available.

READ –

KEY= - Defines the key of the VSAM record to be retrieved. Specifying a key whose length is less than the defined key length results in a generic keyed read in which either the record with a matching key or the next higher key is returned. Keyed reads are maintained separately from sequential file access. For subsequent sequential access use the “POINT” function.

NNNNN – Number of records to place in the data stack. Mutually exclusive with the “KEY=” parameter; Use “POINT” then “READ nnnn” instead.

READU –

KEY= - As defined above.

WRITE –

KEY= - As defined above.

DELETE –

KEY= - As defined above.

POINT -

KEY= - As defined above.

ENDREQ –

No additional parameters required.

Returns

RXVSAM returns a character value of ‘00000000’ upon successful completion of a request. In the event of an error, the return code is set to the appropriate error indication in the following format: nnrrcccc where nn is the RXVSAM error indicator, rr is the R15 value after execution of the failing VSAM operation, and cccc is the VSAM return code of the operation. VSAM register 15 return codes are described in z/OS V1R4.0 MVS System Messages, Vol 6 (GOS-IEA), under message IDC3351I. RXVSAM codes are described in Appendix I.

Technical Description

The program RXVSAM acts as an intermediary stub between REXX and the standalone VSAM access module VSIOMOD. VSIOMOD access is facilitated through the assembler macro VSAMIO, which constructs the

parameter list and invokes VSIOMOD. VSIOMOD is invoked as an attached subtask, and communication is achieved using MVS post/wait logic. VSIOMOD allows access to an unlimited number of VSAM files by constructing dynamic ACB's (Access Control Block) and RPL's (Request Parameter List) for each file. These VSAM control blocks are maintained using an internal mechanism of dynamically created and chained control blocks that exist for the life (Open to Close) of the VSAM file(s) being accessed. Two RPL's are maintained for each file in order to provide independent sequential and random access to the file if needed. This allows for, as an example, programs to read sequentially to the 9th record of a dataset, then randomly access the 40th record, and then continue sequentially reading at the 10th record without the necessity of repositioning for the subsequent sequential read(s). VSIOMOD remains as an active subtask until the last file under its control has been closed, and then it detaches. If all files are not closed before the calling task terminates, a SA03 (subtask not detached) abend is encountered. The following diagram represents the REXX/VSAM interface architecture: