CONNX® Solutions™
Simplified Data Access™

Technology Paper

Using CONNX and Java to Access Data from OpenVMS

by Larry McGhaw, Director of Product Development

Currently, the most popular use of CONNX is to provide enterprise data access to a Windows client PC. However, there are some cases where it would be useful to have this same client access data stored in a VMS database. This can be accomplished with Java. An example of where a VMS driven-connection can be found is when synchronizing a non-VMS database real-time with VMS data.

Java support requires Alpha OpenVMS version 7.1 or greater. The JDK for the Alpha platform is available at Compaq's website: http://www.compaq.com/java/alpha (There is no Java support for VAX machines). New versions of OpenVMS may come with a preinstalled JDK, while older versions of OpenVMS will likely require ECO patches if they have not already been applied. The OpenVMS JDK is similar to JDKs run on other platforms. You must set the Classpath, which under VMS is a logical, to the correct location.

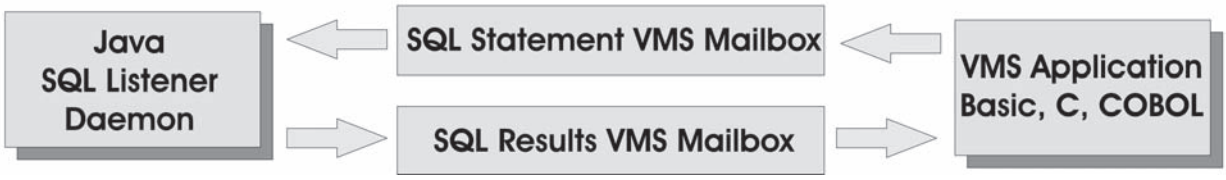
To compile Java source code, use JAVAC.EXE; to run compiled java source, use JAVA.EXE. Most VMS developers program in COBOL, C, or BASIC. How can these programming languages take advantage of a pure Java JDBC driver such as CONNX? One simple way to accomplish this is through VMS mailboxes. A VMS mailbox is a simple queue that can be opened for both input and output. Mailboxes can be accessed using normal file-level semantics, making them easily accessible from all programming languages under VMS.

The first step is to create two permanent VMS mailboxes, using the VMS system call CREMBX. This is straightforward, as the C++ example below illustrates:

```
struct ItemList {
    short     nBufferLength;
    short     nItemCode;
    char     *lpName;
    long      lLength;
    long      lEndOfItemList;
    };

int main()
    {
    int err;
    int nChannel;
    char szDeviceName[256];
```

continued

```
        err = sys$crembx((char) 1, &nChannel, 0L, 100L, 0L, 3L, 0L, 0L);
        printf("CreateMailbox Channel=(%d) Err=(%d) (%d) (%d)\n", (long) nChannel, err, errno,
vaxc$errno);

        // Get the File name for the newly created mailbox
        char      szStat[8];
        ItemList    Item;
        memset(szDeviceName, 0, sizeof(szDeviceName));
        memset(&Item, 0, sizeof(Item));
        Item.nBufferLength = 256;
        Item.nItemCode = DVI$_DEVNAM;
        Item.lpName = szDeviceName;
        Item.lEndOfItemList = 0;
        err = sys$getdvi(0L, (short) nChannel, 0L, &Item, (long *) szStat, 0L, 0L, 0L);

         // Permanent Mailbox created.
        printf("Permanent Mailbox Created: %s \n", szDeviceName);
        }
```



One permanent mailbox can be used for input of SQL statements and another can be used for output results. Two components are required: a Java daemon listener and a VMS application that sends SQL statements. The Java daemon program opens the SQL statement mailbox and waits for input. Once input is received, the program processes the SQL statement and returns the results in the SQL results mailbox. Other applications, written in C, C++, COBOL, or BASIC, open the SQL Statement mailbox and send SQL commands to this mailbox. They then read the output from the SQL results mailbox.

For the complete source to this sample use of Java on VMS, visit ftp://ftp.connx.com/utils/cnxjava.zip

For more information about CONNX, contact:

CONNX Solutions, Inc.
2039 152nd Avenue NE
Redmond, WA 98052
Toll Free: 1-888-882-6669
Tel: 425-519-6600
Fax: 425-519-6601
www.CONNX.com

CONNX® Solutions™
Simplified Data Access™