Advantages of using OpenSCL over scripting language translation

OpenSCL is a software platform designed mainly to assist user's of the ICL/Fujitsu VME mainframe operating system to move their workload to an open systems platform. It does so with remarkable ease.

Migrating from VME involves many challenges including COBOL program conversion, database migration, and migrating online systems. There are numerous tried and tested ways of handling each of these components. A major element of the move to open systems from VME, which is often trivialised, or overlooked is the batch/scripting component. VME's scripting language, SCL, while easy-to-use, is deceptively complex and very often leads to projects of this nature being severely under-scoped and quite often cancelled.

OpenSCL is the only tool of its kind which addresses all the intricacies of SCL.VME differs from other operating systems in that the scripting language, does not only run in interpreted mode, but also in compiled mode, i.e. it can be programmed like any other language. This provides many difficulties for migrators as it essentially means converting an additional language, and since this is usually translated to another scripting language, major functionality is not only lost on the new system, but very often changed. OpenSCL deals with these issues seamlessly by converting the SCL into C, a full and efficient programming language and provides a run-time system to run these C programs as if they were SCL programs.

Mac, File Handling and Procedure Calls

OpenSCL allows an enhanced, fully-functional VME MAC facility on Windows. The system replicates the loading environment and block-structured nature of VME.

All the requisite VME catalogue entities e.g., users, files, libraries, groups, etc., are replicated in the new environment. File handling utilities are supported, including creating of files of any organisation, assigning files, file currencies, filestore maintenance, sorting and printing of any file type.

OpenSCL supports VME file generation numbers completely. Temporary files and file currencies are handled in the same way VME does, i.e. per block. A simple item like this can change the program flow completely as currencies of the same name can exist in successive blocks.

The loading environment is also emulated to ensure programs remain resident in exactly the same way as VME does. This allows COBOL programs to be re-entrant only when needed.

All SCL variables are supported including superstrings and ref parameters. This allows two-way parameter passing between SCL and the called language, e.g., COBOL.

Whenevers

Whenevers are triggers or signals which can be installed on a local or global variable of any type in SCL. Not only is this very difficult to replicate in a scripting language, it is further complicated by the fact that same name variables may exist in different blocks.

ExecuteSCL

VME allows SCL to be executed/interpreted from a string or file of text. OpenSCL supports ExecuteSCL as a literal string or variable as in VME. ExecuteSCL is complex to reproduce as it must include and update the jobspace/environment of calling procedures.

Tapes

OpenSCL replicates the use of tapes on VME. This allows easy backup of files to tape (as opposed to the more commonplace “archive” of open systems) and allows customers to retain their existing backup strategies. Assigning of tape files is not as straightforward as it would appear as VME uses the same commands to assign both tape and disk files.

It is interesting to note that IBM have now introduced tape virtualization on mainframes. This facility has existed within OpenSCL for more than 10 years now.

Journals

As in VME, every action from SCL or COBOL is logged to a journal. This provides a comprehensive audit trail of everything that happens on the system. Even mouse actions which change the user’s environment are logged.

Scheduler, OPS, etc

VME provides various schedulers to control the running of batch jobs and the spooling of output etc. Often these commands are ignored in translation and replaced with available system schedulers. The VME schedulers however allow a programmer to issue commands via SCL to define scheduler operation.

OpenSCL also “fixes” print files which are incorrectly written to by COBOL code.

Editor

In addition to a comprehensive screen editor geared specifically to the development of SCL procedures, a full implementation of the VME line editor is included in OpenSCL. Since these commands are often used during runtime to amend files, it is a vital component to running SCL.

File conversion

Packaged separately, but running on the OpenSCL platform is an automatic file conversion utility. The utility converts files from VME to equivalent target platform types automatically, including translation of the EBCDIC code set to ASCII.

The time saving here alone is tremendous. The alternative here is to convert each file manually.

SCL File IO and User Object Nodes

OpenSCL supports all the VME file handling utilities. It is imperative that each of the features here be replicated exactly on the new platform.

OpenSCL reads and writes all file types - line sequential (an unknown file type on VME), record sequential and indexed files. All sort operations are allowed on the appropriate file types. In addition, OpenSCL allows for IPROC, OPROC and EPROC procedures to be called from each sort.

Both Fselect and Rselect utilities can be specified when using the Basic Utilities System (BUS).

Another seemingly trivial feature often overlooked is the use of User Object Nodes. Many VME installations use these for various control mechanisms, including controlling restart points in jobs. The importance of this feature is obvious. The difficulty in replicating this is that VME can store binary data in a User Object Node.

Speed and Emulation

OpenSCL is often seen as an emulator and hence is perceived to have poor performance. SCL is not emulated, but converted into C code and recompiled on the target platform.

Summary

This document is not a comprehensive description of the features of OpenSCL but is simply to show the advantages of using OpenSCL as opposed to converting the SCL to a scripting language which only caters for subsets of the SCL language.

Often during the project scoping exercise, it is discovered that the new target scripting language may not handle one or two features. It is then proposed that these cases will be handled individually during the course of the migration exercise. This has catastrophic results on project deadlines. It is imperative that all SCL aspects be catered for as each element has a major impact on program flow. For example, the omission of a simple whenever used to stop a program from updating a database can have severe consequences on the data integrity.

OpenSCL is the only product which replicates the VME ED utility and a full sort using IPROC, OPROC and EPROC, as well as the Fselect and Rselect parameters of the basic utilities. If any of these features are not catered for, they must be dealt with individually. In even a relatively small organization, this can prolong the migration exercise by several months.

The advantages of SCL are great. Cost savings are significant. There is no need to buy individual sort packages, new schedulers or new print programs. The timescale of projects are significantly reduced and there are no “surprises” during batch testing. All code runs as it used to in the old environment and follows the same program paths. The risk of error is greatly reduced and companies do not have to retrain staff.