Moving Mainframe JCL to Scripting and Emulators on Azure

Marlon Johnson
Azure Core Engineering Team
Microsoft
August 2022

Contents

Introduction

Legacy applications running on mainframe computers have been at the core of most business operations going back almost 50 years. While these mainframe systems have provided remarkable reliability over the years, they have become somewhat problematic as they can be rigid and, in some cases, hard to maintain and costly to operate.

Many businesses and government agencies (State, Local and Federal) are looking at and planning for ways to modernize these critical systems. They are looking for ways to free up the constrained resources required to maintain these systems while controlling their costs and providing more flexibility when interacting with these systems.

Moving mainframes to the cloud can be a massive undertaking with various systems and utilities programs that must be taken into consideration when planning. There are several Independent Software Vendor (ISV's) with incredible technology that can help move the legacy programs and convert them so that they can execute on x86 platforms on the cloud. In addition to the code that runs these programs on mainframes, there are several other utilities programs and 3rd party applications that either need to be converted or replaced with a similar option that can run in the new environment.

Job Control Language

What is Job Control Language (JCL)? JCL is a scripting language used to communicate with the IBM Mainframe Operating System (MVS, zOS). JCL is used to describe the units of work or "jobs" that run on the mainframes Operating System. It acts as the interface between the programs (COBOL, Assembler, PL/I etc.) that run on the mainframe and the Operating System. The JCL scripts are a series of control commands that establish the criteria for running application programs in batch mode or launching a subsystem on the mainframe.
When batch jobs are submitted to run on a mainframe, the Operating System needs to know how to run the program. Among other things, it needs to know where the inputs for the program are, how to process it, and what to do with the program results or output. JCL had several other utility commands such as COPY, SORT (DFSORT) etc.

When mainframe programs are rehosted, also known as a Lift and Shift, they basically run the same as they did on the mainframe. This requires that the JCL functionality be provided in the new environment for these systems to execute properly.

Scripting

Most modernization platforms and Cloud Service Providers(CSP's) offer both Windows and Linux x86 environment that can host the migrated workloads. The good news is that these platforms have built-in scripting languages that can be used to create scripts that can replace the JCL functionality. In addition to the built-in scripting options, there are also other packages that can be added to the system to provide other powerful scripting languages of your choice. This might be a better option for customers who have in-house expertise for languages like Python or Perl.

Planning

Planning for converting your JCL to a scripting language starts with the Operating Systems (OS) platform your environment has landed on, either Microsoft Windows or one of the many flavors of Linux. While JCL is not a difficult scripting language, it can be verbose, its readability can be cryptic and challenging to someone without mainframe or JCL experience. If possible, the person or team who is writing the scripts should work with someone from the mainframe team who understands how the JCL works.

Language Options

A few of the scripting languages that are commonly available for Windows, Linux, and other POSIX-compliant operating systems are illustrated in the following list.

- KornShell (ksn)
- CShell (csh)
- Bash Shell (bash)
- Tcl/TK

Windows:

- PowerShell
- MS Batch
- Cygwin

These scripting environments are very powerful, and they can provide the necessary functionality to submit and run jobs that were designed to run on a mainframe. As stated above, JCL syntax and structure can be cryptic and difficult to follow. For mainframe programmers shell scripts can also be cryptic and verbose as well. If you are fortunate enough to have someone who understands both, you are off to a good start. If not, the two experts will need to collaborate to decipher and breakdown the JCL and write the equivalent script in the chosen language.

Shell scripting languages are great for text processing, searching, and filtering on data streams and string manipulation using commands such as sed, awk, grep etc. Shell scripts, while powerful, may require writing more code than the JCL equivalent. Depending on what the JCL is performing, be prepared to write many lines of the lower-level scripting statements to reproduce the functionality of JCL.

For mainframe shops that are rehosting their workloads to cloud based Linux and Windows platforms, leveraging shell scripts to replace the JCL code is a viable option. Linux also provides the cron utility that can be used to provide runtime scheduling of these scripts, while Windows has MS Schedular Task.

The best scenario would be that your script programmer also understands JCL. If not, then the script programmer should work closely with someone who understands what the JCL is performing.

Lastly, If you have complex JCL's or do not desire to start from scratch creating the shell scripts, there are a few vendor solutions that can help with converting JCL to scripts. See the following:
Ispirer – Toolkit to convert JCL to Powershell
SoftwareMining
TSRI