TurboHercules

TurboHercules
Mainframe Migration Assistance

This paper describes the major problems facing a mainframe migration effort and the ways that the TurboHercules Migration Assistance System can overcome them.

TurboHercules Inc.

July 2011

Overview

The migration of a mainframe based system to Intel architecture based distributed systems is a complicated and failure prone effort. Even so, many mainframe users are interested in attempting the migration because of the significant cost savings in the final configuration.

The TurboHercules system is a combination of hardware and software that can ease this transition as well as provide a "time capsule" of the original system in case it is necessary to access the original, unmodified software/data at any time in the future.

Contents

1 Mainframe Migration Overview

The typical mainframe based system is large, complicated, usually stable and contains a significant amount of important information (financial, legal, etc). The business logic of the system is contained in a combination of scripts (Job Control Language – JCL), SQL relational database queries and user written code (usually COBOL based). The data itself is contained in a variety of flat files, relational databases (Oracle, SAP, DB2) and hierarchical databases (IMS).

So Why Migrate?

The main driving factor for a migration from the mainframe to Intel architecture based systems is usually cost. For example, in a transaction based system it was found that an Intel architecture based system loaded to the same level as an IBM mainframe was able to service 7.7 million transactions per kilowatt-hour versus 800 thousand transactions per kilowatt-hour with the mainframe. Add to this the cost of ownership of the mainframe hardware and software versus a similar combination with the Intel based system and the case for the migration becomes even more compelling.

Another factor for migration is the steadily decreasing number of people who have the requisite skill set to keep these mainframe installations functional and up-to-date. Mainframe system admins and programmers are retiring or leaving the workforce in increasing numbers. It also appears that IBM is not interested in solving this problem because they are only supporting a handful of schools with any sort of a hardware/software package designed for education.

Another factor is the increasing "brittleness" of the mainframe applications themselves. Many of these applications first became functional 30 or more years ago. Since their inception they have been modified and patched many times, sometimes to the point where the original code is no longer recognizable. Over time documentation has been lost along with bits and pieces of the actual source code. The chances of exactly recreating an application given what is currently available are slim. The current binary object module may be the only instance of the application that is available. This can be a significant problem in banking and other applications where large amounts of money are being processed and an exactly reproducible answer is always expected.

2 Migration Issues

There are many organizations in the business of mainframe migration assistance. All of them rely on minor to major modifications of the data and business logic of the system to accomplish the migration. Unfortunately a significant number of the migration attempts

fail to complete because some small aspect of the conversion is just not possible within the time frame allotted. This is often the result of some missing bit of source code. So, while 80% of the applications have been ported to distributed systems, a small part is stuck on the mainframe until someone has the time and skill to recode it and test it to an acceptable level. We have heard of customers who, faced with this "untouchable" code, simply give up, buy a smaller mainframe to continue executing it and consider the migration a qualified success. Of course, they now have an even more complicated hybrid Intel/IBM configuration, but as long as it is working they are content.

Some of the issues we see are:

1. Multiple interconnected sub-systems that must move off the mainframe all at once. These sub-systems are usually all working on the same data so they all have to move with their data at the same time. This makes the migration more difficult since you have to have all of the sub-systems in a given system converted at the same time.

2. Lack of documentation or verifiable source code. As mentioned before, it is not unusual that a given application has been modified or patched many times of its life time. In some cases the patches may have been a reaction to a particular compiler quirk that had changed from one version of a compiler to another. Without proper documentation and source code, modifications like these would make no sense. It may be necessary to keep the original binary modules in use for an indeterminate amount of time until their replacements can be re-coded and thoroughly tested

3. A need to stay on-line during the conversion. Doing a flash cut-over to a completely new system is fraught with many pitfalls. Even if the initial operation appears to be successful, small, but significant problems may start to show up after a couple weeks of full operation. A gradual cutover allows the full load testing of one new sub-system at a time to make any debugging easier.

3 The TurboHercules Solution

The TurboHercules system is a combination of hardware and software components that make the process of migrating a mainframe system to an Intel architecture based system a straightforward and relatively simple process. When combined with other standard migration strategies it should be possible to accomplish a 100% complete transition in most if not all attempts.

3.1 The TurboHercules Emulator (software)

The TurboHercules emulator is an Intel architecture (x86-64) based package that can faithfully emulate any IBM mainframe from the earliest IBM 360 to the latest z/114. It is

an extended and optimized version of the popular, 10 year old open source Hercules emulator originally created by Roger Bowler. The open source project continues to flourish and many of its members make contributions to the TurboHercules version as well. When it is running in an Intel based environment the emulator provides a configurable virtual machine (down to the serial number of the virtual mainframe) that is indistinguishable from the real thing.

One of the TurboHercules extensions to the virtual machine is a "wormhole" function (patent applied for) that allows the concurrent execution of mixed mainframe and Intel code. Although the performance of emulators is usually anemic compared to real hardware, the combination of 10 years of development and several years of concentrated tuning have resulted in a package that can deliver performance comparable to a 3200MIP mainframe with a single 4 socket Xeon processor based server.

3.2 Mainframe Object Code Processor (software)

For those "orphan" object modules (the ones that can't be replicated due to lack of reliable source code), we have an object code processor that can strip all of the IBM I/O references within the module and replace them with Intel based code that refers to the I/O structure of the TurboHercules system. The resulting code is a mix of mainframe object code (the business logic of the module) and Intel I/O code.

When the hybrid module is executed on the TurboHercules system, it executes as a mainframe throughout the untouched business logic portion of the module then, using the wormhole mechanism, switches to the Intel code to perform any I/O. You can think of the wormhole mechanism as kind of a co-processor in reverse.

Those modules that can be compiled correctly to Intel architecture code can run side by side with their semi converted brothers since the switching from the Intel personality to the mainframe personality is automatic.

Please remember that this functionality may only apply to a small number of the modules in a sub-system, but without it you would have to leave the entire module and the sub-system it is a part of on the mainframe.

3.3 The FICON DASD Interface (hardware)

The overall structure of the IBM mainframe is shown in the appendix. Unlike an Intel based server, the mainframe disk storage (DASD) has always been a separate element in the system. The mainframe cabinet usually only holds the CPU(s) and the main memory.

Since all of the data and applications reside on the disk, it makes sense to have a direct connection to the DASD system from the TurboHercules system. In effect we become another CPU connected to the storage "SAN".

When we combine this configuration with synchronizing software executing on the mainframe itself, it gives us the ability to hand off portions of the job stream transparently to the Turbohercules system. As we convert the original mainframe applications piece by piece, we simply redirect the job stream to the TurboHercules system for each of the converted portions of the application. Since the data continues to reside in the DASD "SAN" no mass movements of data need take place and speed of execution is preserved. Once the last portion of the application has been moved to the new environment, a snapshot can be performed and the mainframe turned off. The Intel architecture system can continue to use the original DASD as a SAN or do a one time conversion to a more cost effective disk storage system if desired.

3.4 The Time Capsule

We have encountered customers who believe that they may have to refer to the original data and or applications at some time in the future. In one case the request to look into an archive was from the government. To provide this capability it is possible to completely encapsulate the original mainframe configuration within the TurboHercules system. The original mainframe hardware configuration, applications and data are contained in a set of files that can be conveniently stored on a backup disk.

If at any time in the future there is a legal request to access this archive (read only of course), the TurboHercules emulator can be invoked using the original machine's configuration. The original operating system can be IPLed (initial program load or boot) and the applications used to access the data as it was at the time the mainframe was turned off.

3.5 TurboHercules Hardware System Cost

The main advantage of the TurboHercules system is the price. For example: a current 2400MIP mid-range mainframe (bare, without memory) costs approximately $2.4M. The monthly maintenance cost is approximately $26,000. The one time price of a TurboHercules system capable of carrying the same computing load in emulation mode would cost $56,000 or a little over 2 months of maintenance on the mainframe.

4 Summary

The TurboHercules approach to mainframe migration is a unique combination of hardware and software. It allows the user to tackle the migration problem in small steps rather than giant leaps. It is a lot easier to keep the overall system up and functional as portions of the application are moved from the mainframe to the Intel architecture distributed systems. In the unfortunately not so rare case that the correct source code for

a portion of the application cannot be found, it is still possible to employ the original binary object module until a new module can be created and tested thoroughly.

As a bonus, using the Time Capsule function, a snapshot of the complete original system can be taken just before the mainframe is turned off for the last time. In required by law, this snapshot would allow the user to access the archived data at any time in the future.

Appendix

Equipment Configuration