Cloud Transformation and Automation Group: The Realities of Mainframe Application Migration

Thomas Klinect

What's The Goal or Endgame

When the mandate to integrate the Cloud into the current landscape or to move everything to the Cloud drives the corporate mindset, that same mandate rarely includes the realities of such a move. We typically consider the following the promises of Cloud:

- Reduction in current IT costs
- Ubiquitous access
- Always available
- Faster innovation
- Reduction in IT and technical debt

Often, organizations fail to consider the costs to even move a couple of applications into a public Cloud environment, let alone their entire corporations.

While considering the listed items in a transformational program, companies must remove the marketing spin placed on each of the listed items by Cloud providers. Cloud providers do not know how your organization uses technology. Your current technology stack is a mystery to Cloud providers, therefore their promises lack the realities of your business.

Understand the goals you are attempting to meet with the new program. If this is just a board-level directive, push back to better understand why and even if the organization needs the proposed changes. Provide a view of the business effects of any changes in the IT environment. Quantify each impact with financial information not from the Cloud providers, but rather from your organization's day-to-day operations. Will moving to Cloud or even a Hybrid Cloud truly provide the benefits outlined?

To provide this level of information, consider the following:

The Promise of Public Cloud and the Realities

The Cloud sales pitch is simple: "Only pay for what you use." But the promises do not stop there:

- Availability is another promise of Cloud, as you are always on.
- Clouds are on the latest technologies and therefore when you move your applications to Cloud, you'll have the latest technologies powering your business.
- Changing technology to the Cloud is as simple as plug-n-play.
- Software upgrades and patches are no longer a concern.

- Hardware obsolescence is no longer a concern for the IT department.
- IT departments become “ticket takers” as the Cloud provider manages everything.

The Reality

Taking all the promises and benefits into consideration, if the Cloud movement started in 2006 and we fast forward to 2021, then why is every corporation not completely on the Cloud today? If you only pay for what you use and the Cloud is “always available” and your organization becomes technologically advanced through the use of Cloud, what is preventing the adoption? With 15 years in the rearview mirror, we can now see the Cloud does not save a lot of money and getting there is extremely hard, time-consuming and expensive.

The Grass Is Not Always Greener

The reality is each promise mentioned above contains some truth, but they are far from the complete truth. Yes, you can save money, but not in the way you may first consider. Your applications have to run and they have to run somewhere. This means they will consume resources, either yours or the Cloud providers’.

In reality, there are very few applications which only run for a portion of the day. And those which run only occasionally rarely consume much of the existing resources within the IT department. Applications typically do not run singly. Normally, many applications run to execute business processes. So, moving a single application to the Cloud means you have to move all the associated applications, and the associated data stores to the Cloud. Splitting business process between on-premise and Cloud complicates everything from design to support. Which adds even more hidden costs to the bottom line.

Most corporations use applications which run in many operating environments such as UNIX (and many variants like HP-UX and Solaris, to name a couple) OS/360, VxWorks, pSOS and other heritage operating environments implemented over time. Even when the application platform is already Cloud-based, companies built their applications using older technologies which may no longer exist today. What does an enterprise do with these platforms? A company can’t simply abandon them as they are part of the business process and the applications need to remain in the business process flow in some form.

There are many ways to deal with this:

- Emulate the environment
- Rewrite
- Re-platform
- Re-architect
- Replace

Yet each method has a cost associated with it. Not just the cost to change the application, but the costs to re-integrate the application into the enterprise again. Ordering the technologies from least to most expensive puts re-platforming at the top of the list as least expensive method of moving an application to a Cloud platform through recompiling the code. An organization ends up with the same application with the same bugs, only now it is running on another platform. A simple example of this is to recompile COBOL from a current platform to the target Cloud platform. Of course, very few applications can simply be recompiled as they use platform specific functions which change during the recompiling. So even in the least expensive case, there is still a cost to move the application.

The most expensive method to move existing applications to a Hybrid Cloud platform is through a rewrite, which requires more business knowledge than technology knowledge. The development team must understand *what* the application does and how it accomplishes the this within the bounds of a new (more modern) language. Yet when the rewrite is completed, the amount of testing required drives the costs and delivery dates further out than any development team would like to admit.

However, understanding the business rules embedded in the code is key to a rewrite. Extracting these rules is very problematic and requires external tools and in-depth knowledge of the business operations to make the extraction correctly. Yet the costs to extract the business rules, create user stories, then develop and test each component add additional cost to moving the application.

Understanding the business rules of the application actually applies to every other form of modernization mentioned in the list above. Which means every method has a shared cost of business rule extractions, yet the biggest impediment to moving any application—through any modernization process—is the enterprise complexity analysis (ECA)—or more simply: how does each application interact with every other application in the enterprise.

Think of ECA as a pot of cooking spaghetti. When you reach into a pot of boiling spaghetti to determine if the spaghetti is ready, you typically grab a strand of spaghetti and lift it away from the pot. If the strand comes away with few other strands attached, the spaghetti is ready for serving. This same paradigm applies to understand the complexity of an enterprise. You must identify an application and determine how many other applications depend on it or which applications the selected application depends upon. This is the basis of ECA.

If applications were simple, this would be a straightforward process; however, applications are highly complex and developed over time, which makes the complexity of any single application very high. Now extrapolate that complexity to the thousands of applications in an organization and you begin to understand the daunting view of moving any application through any modernization method.

The list above then expands to the matrix below. Each method uses a unique combination of tooling to assist in the transformation of that application.

Method	BRE	ECA	Dev	Test	Knowledge
Emulate		$$$	$	$$	$$$$
Rewrite	$$	$$$	$$$$	$$$$	$$$$
Re-platform	$	$$$	$$	$$	$$
Rearchitect	$	$$$	$$$	$$$	$$$
Replace	$$$	$$$	$	$	$$$

Caption: Cost considerations for application modernization

But the costs don't stop there. Availability is often overlooked when moving from a mainframe environment—where every application enjoys five-nines of availability—to the Cloud, where three-nines is normal. This should concern an organization. Can your business endure an application outage with no accurately defined period of recovery?

Do not confuse availability with accessibility. An application is available when it is running and awaiting interaction. However, if a client or user can't connect to the running application, it is not only NOT accessible, the application is also NOT available. In traditional Cloud contracts, you will find the phrase "best effort" when discussing SLAs. If the Cloud provider can't guarantee access to your application, then you have lost more in the transformation than any organization should relinquish.

More Failures Than Success Stories

One organization which was successful in removing the mainframe was Swisscom. Swisscom moved 2,500 MIPS to the Cloud, however that move took more than 2.5 years.[1] From all published material, it appears Swisscom moved their application from COBOL on the mainframe to COBOL in the Cloud. All nicely packaged in containers, but it's still COBOL! One reason companies move applications from the mainframe is because of technology or resource issues such as COBOL skills. Moving COBOL from one platform to another does not solve the technology issues, it simply moves the location of the problem. But, had Swisscom transformed their mainframe environment into another language, the cost would have been significantly higher.

Yet, for every success story out there, there are thousands of failures. Most organizations do not publicize their failures as this infers a weak business when in reality it is just poor planning and execution of the modernization.

A company in EMEA convinced their board of directors to spend USD 90 million to remove all mainframes from the enterprise. After years elapsed and all the money was all spent, an analysis of the enterprise showed only a two percent reduction in the mainframe footprint.

[1] "Swisscom Moves Entire Mainframe Workload to Software Defined Mainframe in the Cloud," https://www.lzlabs.com/swisscom-moves-entire-mainframe-workload-to-software-defined-mainframe-in-the-cloud/

A postmortem of the program showed that a lack of enterprise understanding caused the failures, thus shelving plans to remove the mainframe from the enterprise. This scenario plays out more across the globe than any other form of mainframe modernization.

Another somber moment in mainframe removal programs was for a US telecommunications company, which removed the mainframe and replaced it with a new private Cloud. It took over five years with millions of overruns before the team accomplished their goal and removed the mainframe. The platform they implemented in their data center was a mere shell of the original platform, although it was complete enough for the telecommunications companies' users.

The Mainframe Continues to Be a Good Option When Leveraged Correctly

Mainframe costs have reduced over the last five years and with tailored fit pricing programs such as the one offered by IBM, customers can grow workload activity on an IBM z15 mainframe with lower TCO and OPEX costs than on a public Cloud. Open-source software has also found its way onto the platform which allows for innovation—just like that seen on distributed platforms. Companies can now implement DevOps on the mainframe with the same tools their distributed counterparts have used for years. By embracing DevOps on the mainframe, existing mainframe developers can expand their knowledge to other platforms and more advanced tools to optimize their development efforts. Distributed developers now have another platform to develop applications on since the development tools used in DevOps are on a par in both environments.

As an example, State Farm implemented DevOps on the mainframe and their ability to develop applications is now on par with distributed platforms.[2] The State Farm software teams now develop at the pace of the customer and not at the pace of technology.

"By bringing agile DevOps practices to IBM Z we will continue to accelerate development cycles ultimately delivering new services to customers faster so that we continue to meet their expectations."

— Krupal Swami, Technology and Architecture Director, State Farm

The difference between running in the Cloud and on the mainframe is more semantics than it is a platform question. If an organization moves from the mainframe to a Cloud running a Linux operating system, the company can apply the same application modernization methodologies to move their applications in either an IFL or a Linux container to a mainframe running Linux.

The organization must still make the same application changes to conform mainframe COBOL to Linux COBOL. Yet, as mentioned previously, the enterprise will still be running COBOL and

[2] "Combining the speed and agility of DevOps with the robustness and security of enterprise servers," https://www.ibm.com/case-studies/state-farm-systems-hardware-devops-agile

not a more modern language. Separating applications from the mainframe to a public, private, or Hybrid Cloud introduces latencies to modernized applications as there is now a network layer inserted between mainframe applications and the Cloud applications. Yet, running Linux on the mainframe removes most latency issues, since there are software-based network connections between the Linux application and the mainframe host.

With all the changes implemented on the mainframe over the last ten years, the mainframe is now a viable platform for distributed Linux and UNIX developers and testers. As IBM continues to change the way the platform is run, the mainframe will continue to be a viable platform for years to come.

Side Effects of Modernization Which May Be Overlooked

It's not just the technology which makes a modernization so complex it's the hidden costs which quickly emerge after a deployment. Some of the non-obvious questions one should ask before beginning a modernization program of any type are:

- Do you have the current staff to run the proposed system?
- Does introducing new staff disrupt the business?
- What happens to all the corporate knowledge when you shift to another platform?
- How long does it take to recover from a technology interruption?
- How will the end-users of the new technology handle the change?
- Can the development shop make the leap without increased expenses?

As an example: The Internal Revenue Service moved several programs from older UNIX platforms to newer Windows platforms in the early 2000s. The technology used was a POSIX subsystem in Windows. While they moved the applications using several modernization methods, the applications were in the same language (re-platform). Now they needed to find UNIX programmers to maintain and enhance the application on the Windows platform.

The IRS rapidly executed the deployment of the new application without deep consideration of how this change would affect the end users. The IRS provided some training to their staff; however, the switch from green screen to the mouse was more than most employees could manage.

To make up for the technology deficit, the IRS ended up hiring temporary workers to keep the taxpayers' queues low. This resulted in millions of dollars being spent to recover from deploying new technology into the field.

Conclusion

You can move applications from one platform to another, however, the cost and complexity *may* out-weight the value of doing such a move. Right-sizing, on the other hand, makes complete sense—not only when considering a mainframe, but when looking at distributed platforms as well. Only remove processes from the current platform which would run better on another platform while enhancing the remaining applications on their current platform using new technologies which have been deployed.

Don’t let the hyper-scalers convince you to move everything from your platform to their platform. While the tools and the process of re-architecting applications are better, there is a significant cost to move applications off any current platform. The only one who wins when the hyper-scalers get involved is the hyper-scalers themselves. This is because once you move your application to their platform, it is very hard to return to the original platform if the software fails (under excessive load, for instance).

Before considering a transformation or if you are amid a transformation: **STOP**. Stop and consider:

- What are you attempting to gain from the transformation?
- On which platform will each application (including sub-applications) run best?
- Will you really save money or just reapportion it into another category?
- How many applications can change from running 24x7 to only when needed?
- How will the change affect your organization, including:
  - Staffing
  - Licenses
  - SLAs
  - Subcontractors
- What is the effect of changing from perpetual licenses to subscription?
- What is the cost factor of all application environments, including:
  - Dev
  - Test
  - Production
  - High availability
  - Disaster recovery
- Can your business processes handle the latency that moving applications introduce?
- Can your existing applications work where latency is non-deterministic?
- What is the definition of *available* for each business process?
- What is the true cost for Cloud (down to the instruction level), including:
  - Disk storage
  - I/O
  - CPU
  - Network
    - Including the new links to the hyper-scaler
  - Backup
- How is security handled by the hyper-scaler?
- How do you change the current security profile to handle bidirectional network traffic?

Remember, it’s never an all-or-nothing proposition.

About the Cloud Transformation and Automation Group

CT&A assist clients transforming from their current environment into any another environment. The target and source environments are typically a mix of operating systems and supporting hardware, which complicates any transformation. CT&A is uniquely qualified to advise clients using their real-world experience and leadership across all platforms to ensure success.

This document was developed with IBM funding. Although the document may utilize publicly available material from various vendors, including IBM, it does not necessarily reflect the positions of such vendors on the issues addressed in this document.

November 2021