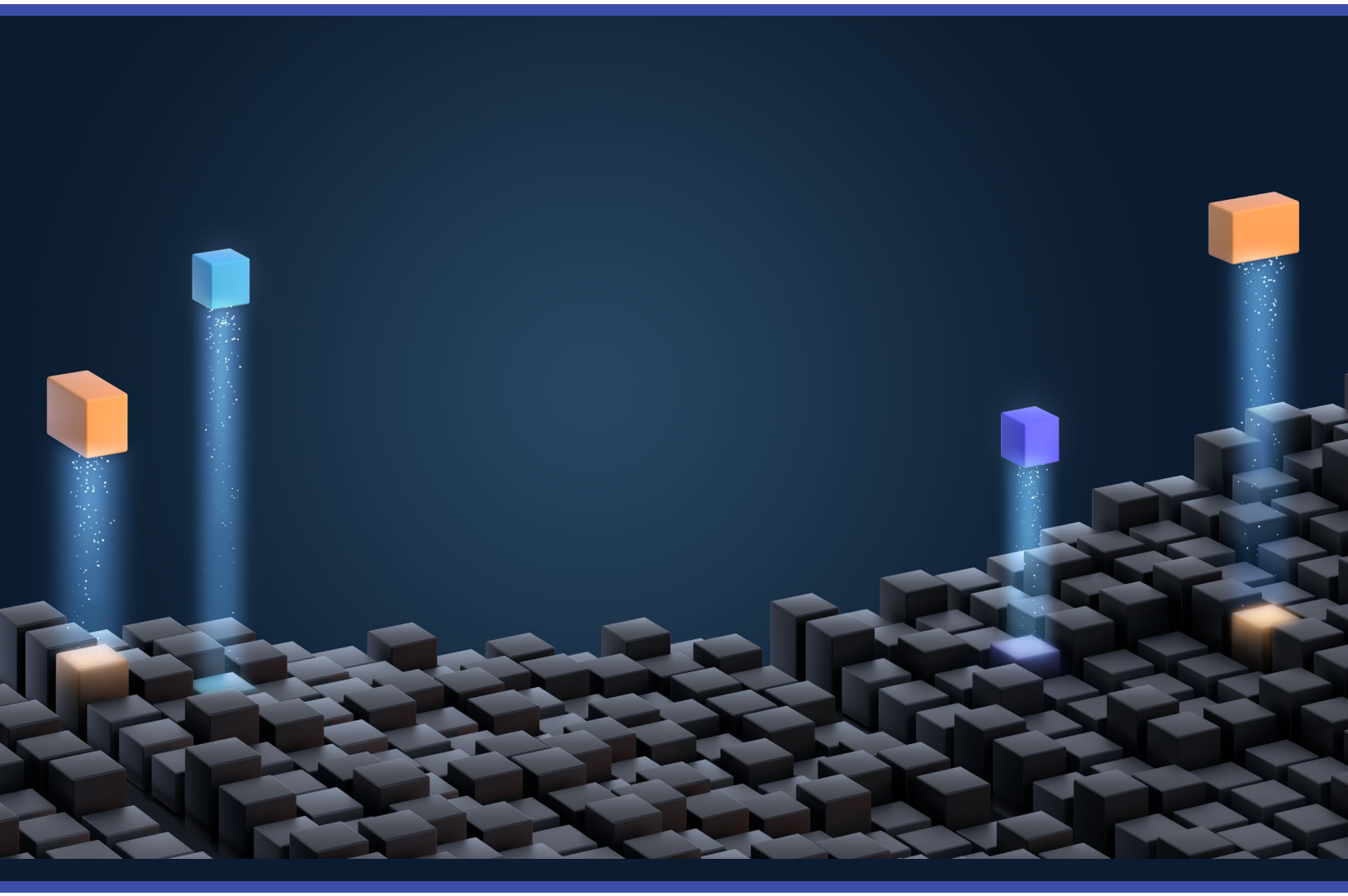
OpenLegacy
Digital to the Core

White Paper

How your peers solved common IBM mainframe challenges

Featuring microservices migration, middleware, global APIs and VSAM

By Hans Otharsson, Customer Success Officer, OpenLegacy



www.openlegacy.com

01

The journey from mainframes to microservices

The journey from mainframe monolithic applications to microservices is usually "fraught with peril." Many of you may find yourself struggling against technology gatekeepers and bureaucrats, all trying to preserve the past with no eye to the future. You may also need a machete of steel to hack through the forest of integration layers. The good news is that other companies have addressed these issues head-on, found their path and lived to talk about their successes.

For example, a major bank found themselves in that "fight for their lives" scenario—at one point they were the king of the hill, able to provide the latest products and services to the demands of the day. The problem was that the demands of the day kept changing, and they had no capability to keep up. So, they started to slowly step down that hill, which over the years turned into a rather fast roll.

They knew they needed to change; they hired the best and the brightest. They began building the best of breed new products and services. But what the bank quickly realized was they did not operate in the "system of engagement" realm, they operated in the "system of record" world. These two systems did not play or talk nicely to each other, therefore setting up an integration flow meant these two respective teams needed to understand the needs of the other. The communication was difficult because one side kept talking in the language of security, uptime, transactions, and other monolithic terms, while the other side kept speaking in a dialect of time to market, innovation, flexibility, open standards and automation.

This bank overcame the technology barriers and the corporate culture barriers by using OpenLegacy to enable their "Legacy Integration Infrastructure Modernization."

For the longest period, this bank suffered from the burden of the countless layers of integration solutions purchased or built in-house. These numerous layers were all incorporated with the best intentions to solve an immediate business need or as part of an enterprise wide strategic architecture plan. But the reality was when something new was brought in, it did not replace the old, and instead was built on top of the old. So, they did not just end up with numerous layers, they had interdependencies between layers. To further compound the complexity and brittle nature of these integration layers, they replicated this problem across their different back-end technology hardware environments (Mainframe, iSeries, Tandem).

So why was this a problem? It was a problem because when the bank needed to create a new product or service, they needed to navigate these tightly bundled, co-dependent integration layers to finally reach their core systems of record. This navigation was a complex journey; it was a journey needing to find the right resource with the right skill sets and the time to accommodate the communication at each layer. This process was risky and prone to delays and budget overrun due to dependencies on resources, technology, priorities and expensive skill sets. Does this sound familiar to you?

The solution involved an API Factory which bypassed decades of vendor and homegrown integration layers. Built on a foundation of automation, standardization, and open standards, the bank's new API Factory enabled them to be a competitive market leader in the roll-out of new products and services. Furthermore, it set the foundation for future projects, enabled API reusage and helped align the culture around the common principles of the API Factory.

Built on a foundation of automation, standardization, and open standards, the bank's new API Factory enabled them to be a competitive market leader in the roll-out of new products and services.

02

Do we still need middleware? If so, where?

The topic of middleware is both broad and deep, covering many varying products and platforms. For the purposes of this discussion, we are referring to Message Oriented Middleware (MOM). The top 4 MOMs being: IBM, Oracle, Microsoft and TIBCO.

At a high level, middleware software finds itself between a front-end client (mobile device, web browser or any agent that requires services) and a back-end application (often legacy applications) that supplies the requested information.

In simple layman terms, middleware functions much the same as plumbing. It provides the pipes, joints and fixtures to facilitate the throughput of data. But is it still needed? Yes, plumbing is still necessary, and middleware still plays a role in the flow of real-time information access within an enterprise and between enterprises. Instead, one should ask "is traditional middleware the right tool for the digital economy?"

MOMs made sense when we had dumb endpoints, but now with microservices and serverless functions, the endpoints are smart. That means smart middleware just slows the connection down and leads to maintenance headaches. This is the reason most modern development paradigms use the mantra "smart endpoints, dump pipes" and look to limit their reliance on MOMs.

Once you make the future-thinking choice to leverage modern architectures like microservice-based APIs and a "smart endpoints, dump pipes" philosophy; you need to put the data conversions in the legacy endpoint. The question is how to handle this without modifying the legacy system? The answer is simple: use automated code generation to handle the conversion and incorporate it into the endpoint outside the legacy system. The code generation piece is critical, since each type of legacy data needs a specific data conversion to create a fast, direct connection.

This is the reason most modern development paradigms use the mantra "smart endpoints, dump pipes" and look to limit their reliance on MOMs.

For example, a major retail organization paid a hefty price for acquisitions, international regionalization and departmental IT silos. These factors pulled a limited IT budget into numerous directions and spawned a hodgepodge of solutions, technology and skill sets. The eureka moment for this retailer was when they realized that they could no longer thrive without an online retail presence and just surviving was not an option.

They had a basic online presence, one that did very little more than "help me find a store" or "download a coupon." They needed the capability for customers to create a profile, link that to a customer loyalty program, create a shopping cart, browse their online store, make suggestions for out-of-stock items, choose pickup or delivery options and return or exchange merchandise.

From a front-end development perspective, this functionality was straightforward. The problem was integrating several different internal systems across a variety of hardware platforms. A microservices DevOps approach automated the exposure of the legacy application's functional endpoints, and enabled the front-end development teams to consume the appropriate and highly anticipated microservice. These microservices, free of complexity and barriers, were the necessary building blocks of new features to their online presence. This direct path to the system of record bypassed the various MOMs, enabled a faster development lifecycle, simplified testing requirements and provided the front-end developers the service they needed.

03

Global APIs: Integration that crosses borders and technology stacks

Organizations often require a single view over the vast array of application technology silos running their business operations. Having a variety or disparity of technology stacks is often a symptom of mergers and acquisitions, or the consolidation of regional offices. Regardless of how they got there, it is a problem that impacts the three pillars of an effective API strategy (revenue growth, customer experience & operational efficiency).

For example, a major global bank had to take care of the operational and regulatory reporting requirements needed to operate in 13 different countries. Much of this work was done through the adoption of a new 3rd party system and a labor-intensive regional process to keep everything in sync.

The bank initiated a Global API project to mitigate the regional disparity in the customer experiences, and to immediately deal with the dramatic changes in services a client would need to access as they traveled and banked in various countries. The decision began with three lines of business (checking, savings and credit cards) requiring consolidation of 11 different applications across the countries. But digging deeper, they realized that some of the regional offices had heavily customized their deployments and they all seemed to have different ideas on account numbering and identification schemas.

The objective of the Global API was to create an integration layer that provides a consistent way to interact with the underlying regional legacy applications. Let's look at money transfer as an example: say a client wants to transfer money from Account 1 to Account 2, but Account 1 and Account 2 are in two different countries, running two different core banking systems. The job of the Global API is to access to the appropriate API, orchestrate the call to subsequent regional API's, and perform the necessary message data mapping for the respective business function.

With the Global APIs, the product roll-out schedule is not dependent upon each regional bank. In the past, it was like rolling out 13 new products and services. In this new paradigm, the developers simply build the new products and services on top of the Global APIs, without needing to impact, change or even interact with the regional operations team. Now new products roll out across all locations with all customers having the same level of high-quality experience. New products and services deployment cycles are reduced by a factor of not just months, but years.

With the lessons learned from this experience, the bank has now started a parallel initiative to expose each regional bank's general ledger and compliance systems to this Global API structure, removing manual and error-prone tasks.

04

Rights and regulatory mandates as business drivers for innovation, despite VSAM repositories

Historically, all levels of government have a considerable impact on an enterprise's IT environment for things like labor rates, tax rates, and reporting. However, the advent of rights and privacy-related regulation—like the GDPR in the European Union, CCPA in California and HIPPA—add a new dimension of complexity.

These "Right to be Forgotten" regulations require that companies have a true 360° view of their customers including every connection point, no matter how old or insignificant. This is not a simple ETL (Extract, Transform, Load) solution, because a snapshot of data is not enough. This situation requires an approach

that continues to update as changes are made within the core systems.

For example, a major insurance company found themselves with databases buried deep down in their mainframe programs and other legacy systems. Some systems only get accessed through a COBOL application using a VSAM file while other systems use different datastore mechanisms. All of these legacy data records need to be combined with modern cloud-based databases found in newer applications spread throughout the organization.

So, the needle in the haystack exercise was to pick through these archaic data stores and archives, as well as modern databases, to find and validate all references to the specific person or group of people. This was a painful, time consuming and error-prone process.

The insurance company easily identified a tool for analyzing their multiple cloud-based databases in the search for the correct records. The struggle was accessing and updating the legacy data stores.

In the process of assessing their respective data stores and application processes, the company reached the painful realization that over 65% of the relevant customer personal data was stored and maintained within their rather extensive VSAM data repositories. This presented a major speedbump, especially if it required the internal IT teams to begin writing query and extract programs to not only the production environment, but also their backups and archival data stores.

This company found a way to “know their data” and “access all their data” by following these steps:

- First, provide a mechanism for their Data Privacy Platform (DDP) to perform the necessary search of their legacy data stores. The ability to seamlessly interact with the VSAM data stores as part of the DDP interrogation process with standard Restful API’s was critical for consistent assessment and reporting.

- Second, once a relevant record was identified, a set of well-orchestrated APIs validated the record along with identification of any associated records, including records in backups and archival data stores.

- Finally, instead of writing special, one-off programs to delete the VSAM data stores, they just changed the API to handle this process. Using the API for deletion reduced their technical debt.

If you have IBM mainframe challenges, OpenLegacy might be a good fit. Please visit us at www.openlegacy.com.

About Hans Otharsson

Hans is a global leader in legacy transformation programs with decades of experience developing, enhancing, maintaining, troubleshooting and transforming so called 'legacy applications' and associated environments. His global journey has taken him into countless environments and business scenarios, where his "straight to the point" approach has enabled him to bring true change and business driven transformation to his clients. His ability to quickly assess a situation and determine if an organization can bring value—and offer suggestions for other alternatives if needed—has made him a trusted advisor to numerous global organizations. Hans has many years' experience with 'legacy modernization' in senior executive roles at Consist Software Solutions and Ateras. At Software AG, Hans was responsible for Professional Services Sales & Delivery in North America and Canada. He also founded ModernWiser, a consultancy helping organizations understand their legacy modernization options. Hans started his career in banking and enterprise accounting systems.

In his current role as Customer Success Officer at OpenLegacy, Hans is responsible for corporate operations and client success. These dual roles capitalize on Hans' strengths of quality delivery, a customer first mentality, and solid industry experience—all of which are truly echoed in his mantra: "We measure our success on our clients' success."

About OpenLegacy

OpenLegacy's Digital-Driven Integration enables organizations with legacy systems to release new digital services faster and more efficiently than ever before. It connects directly to even the most complex legacy systems, bypassing the need for extra layers of technology. It then automatically generates APIs in minutes, rapidly integrating those assets into exciting new innovations. Finally, it deploys them as standard microservices or serverless functions, giving organizations speed and flexibility while drastically cutting costs and resources. With OpenLegacy, industry-leading companies release new apps, features, and updates in days instead of months, enabling them to truly become digital to the core.