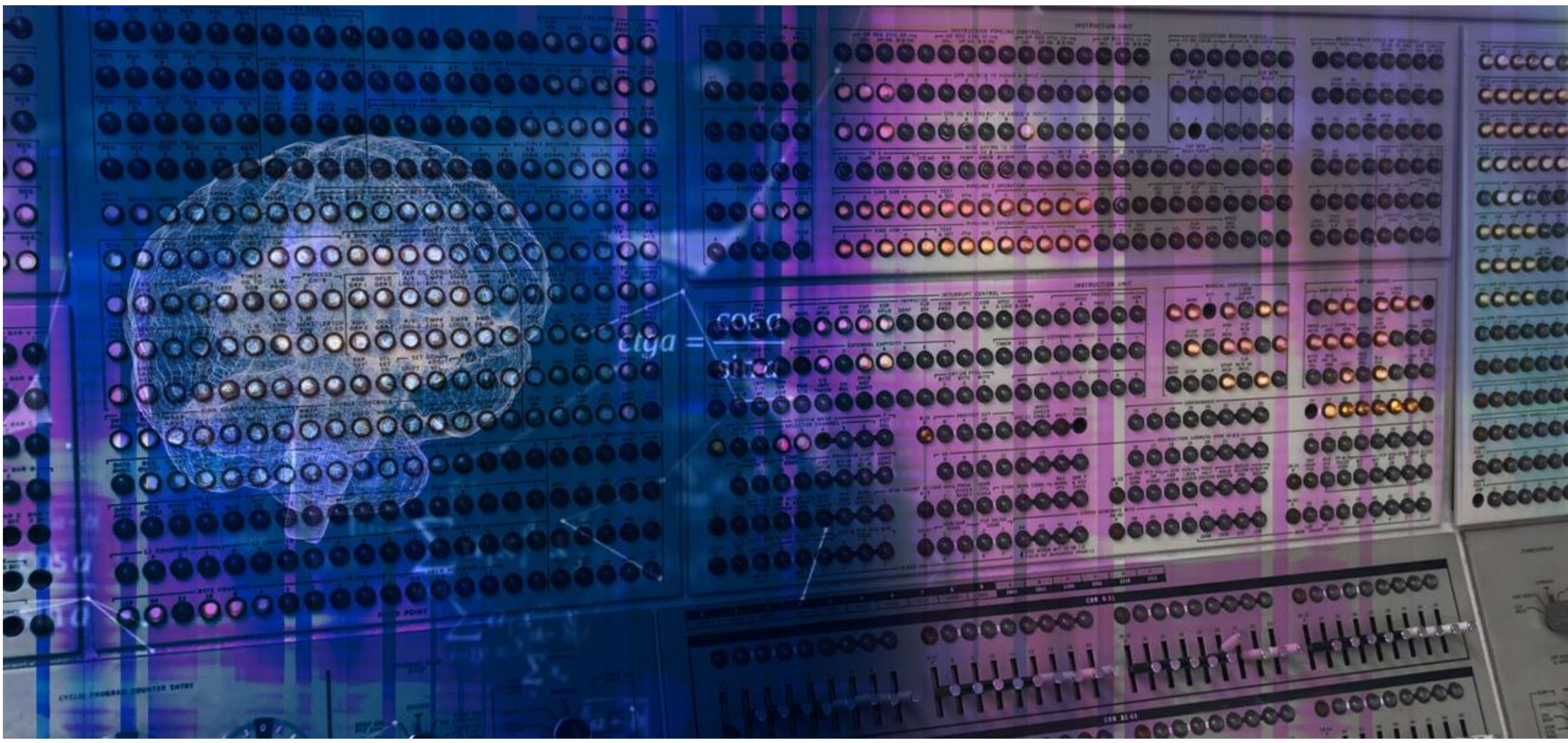


White Paper

Legacy Technology Got You Down?

Try Microservices and Cloud-Native Architecture

Jason Bloomberg
President, Intellyx

September 2019

For any enterprise with legacy IT assets, modernization is both an essential business priority as well as a risky, expensive endeavor.

With the rise of hybrid IT and cloud-native computing, enterprises now have far more powerful tools in their modernization toolbelt than they had a mere decade ago.

Modernization, however, doesn’t live in a vacuum. As enterprises move to the cloud, they increasingly implement a mix of cloud and on-premises environments we call hybrid IT.

Essential to the hybrid IT modernization story is the increasing popularity of microservices, a key component of the cloud-native approach to enterprise IT that brings the benefits of cloud computing to all aspects of IT – including modernization itself.

Implementing microservices as part of a modernization strategy, therefore, is an essential enabler of this cloud-native vision for IT, and OpenLegacy offers the capabilities necessary to help any organization with its modernization challenges as it moves toward this vision.

Transforming the Context of Legacy

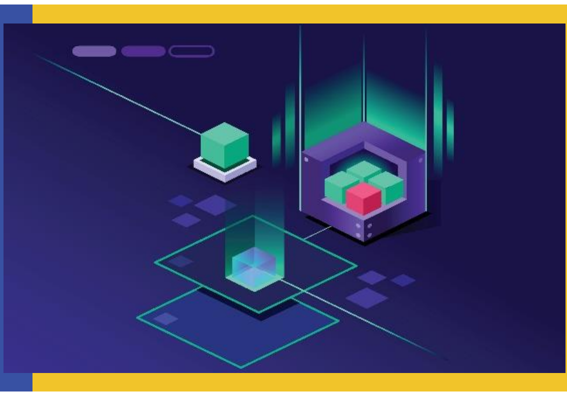
No word sends chills down the spine of a CIO like *modernization*. Legacy technologies can themselves become money pits, but nothing compared to the modernization initiatives of old – vast, risky, and outrageously expensive. Worst of all, most modernization efforts simply failed to deliver on their promises.

Since the first computerized enterprises replaced their aging vacuum tube systems with newfangled solid-state technology, modernization has always been expensive and risky. Over the years, as one generation of hardware and software replaced another, IT executives continued to struggle with the core dilemma: leave older technology in place and make do, or take the riskier path of rip and replace.

This dilemma invariably led to the increasing prevalence of *legacy* – essentially obsolete technology that nevertheless remains in place because it still provides value, and IT leadership deems replacing it as too risky.

In those days, legacy modernization was a black or white affair: either rip out all the old technology and start afresh, or make do with it, layering new technology onto old to extend its useful lifetime. Afraid of the risks of the first option, most IT executives opted for the second, for better or worse.

Legacy modernization was a black or white affair: either rip out all the old technology and start afresh, or make do with it, layering new technology onto old to extend its useful lifetime.



Today, two epic changes have transformed this dichotomy: first, technologies have fundamentally improved, offering IT executives a wider range of choices. And second: digital transformation priorities have upped the ante on legacy modernization.

In the digital era, enterprises simply cannot afford to keep much of their aging legacy technology around. But falling into the older, black and white way of thinking about modernization is a false solution.

In addition, the IT organization and its role within the enterprise has also transformed. IT is no longer simply a cost center in the enterprise. In the digital era, software has become strategically important as companies reinvent themselves as technology-driven organizations – thus raising the bar on modernization.

No longer can such forward-looking enterprises skewer themselves on the legacy dilemma. There has to be a better way of looking at modernization.

How Did We Get Here?

The core modernization dilemma – whether to leave legacy as-is or rip it out – invariably led to the increasing prevalence of legacy. Over the last twenty years or so, one innovation after another has chiseled away at this legacy challenge.

In the early 2000s, Web Services brought loose coupling to enterprise integration, easing the task of replacing individual components in complex, interconnected deployments.

Toward the end of the decade, REST proved lighter weight and easier to adopt than Web Services, greatly simplifying the task of modernization.

In conjunction with Web Services, we deployed *service-oriented architecture* (SOA), whose implementation typically depended on sophisticated middleware. These enterprise service buses (ESBs) handled a variety of tasks, including integration, routing, data transformation, security, and more, while exposing application functionality typically as Web Services.

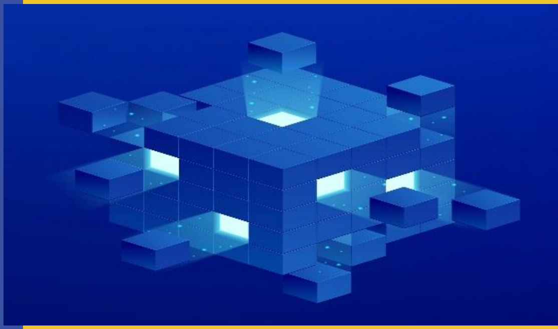
SOA was therefore able to expose lightweight, language-independent service endpoints by shifting the intelligence to the middleware – a pattern we now like to call 'smart pipes, dumb endpoints.'

At the same time, virtualization came to the fore, bringing a comprehensive abstraction layer that isolated software challenges from the underlying hardware. Leveraging this technology, it often became possible to separate the question of hardware refreshes from the equally challenging discussion of software updates.

Then the cloud came along, and everything changed. As the result of the cloud transforming the role and nature of middleware, coupled with the virtualization-driven rise of containers and microservices, SOA eventually gave way to *microservices architecture*.

Unlike Web Services that were little more than 'dumb' XML-based endpoints, microservices are cohesive, parsimonious units of execution (that is, as small as possible but no smaller) – little packages of goodness that only do one or two things, but do them well.

Microservices are cohesive, parsimonious units of execution (that is, as small as possible but no smaller) – little packages of goodness that only do one or two things, but do them well.



In common parlance, we refer to microservices architecture as 'smart endpoints, dumb pipes,' because the microservices are their own mini-programs, with all the smarts we can cram into them. But to integrate them, we typically use nothing more intelligent than HTTP-based RESTful interactions or lightweight, open source queuing technology (in other words, 'dumb pipes').

Replacing ESBs with 'dumb pipes' made sense in the context of the paradigm shift from SOA's on-premises context to the cloud-centric world of microservices architecture, but implementation, scalability, and agility challenges remained.

With the rise of microservices comes another relatively new trend: modernizing elements of complex, distributed applications while leaving other components alone – or more generally, updating different parts of such applications on different schedules, depending upon the needs of the business.

This modular approach to modernization, in fact, usually depends upon microservices. Microservices have become the keystone that links legacy assets and modern, cloud-native IT infrastructure. However, even for organizations that are still in the early stages of adopting microservices architecture, they can be an important part of the move to hybrid IT.

How Hybrid IT Is Reinventing both Legacy and the Cloud

The legacy modernization challenge has always boiled down to an economic argument: how much is an out-of-date system costing the organization, vs. the all-in cost of modernizing that legacy – including all the indirect costs of making the transition from old to new, including downtime, retraining, customer resistance to change, etc.

The rise of *hybrid IT* has thrown a new option into these traditional economic arguments.

Hybrid IT is a workload-centric management approach that seeks to abstract the choice of deployment environment across multiple public clouds, private clouds, and on premises and cloud-based virtualized environments, as well as traditional on-premises systems, including legacy assets.

Hybrid IT is a workload-centric management approach that seeks to abstract the choice of deployment environment across multiple public clouds, private clouds, and on premises and cloud-based virtualized environments, as well as traditional on-premises systems, including legacy assets.

As IT leadership realizes that the cloud is no panacea, and that a cloud-first strategy doesn’t mean cloud-only, the hybrid IT alternative is coming to the fore.

However, hybrid IT does not mean resigning oneself to an inherently heterogeneous technology landscape. Rather, it is an intentional decision to have a mix of both cloud and on-premises environments as appropriate to meet the business need.

Today, hybrid IT is rapidly becoming the overarching paradigm for modern IT operations. However, in spite of the inclusion of legacy under hybrid IT's umbrella, we don't mean for hybrid IT to *perpetuate* legacy. Rather, hybrid IT gives enterprises the means for *modernizing* legacy within a new context.

This new context recognizes that there is far more going on with today's modernization than in the old days of the rip-and-replace dilemma.

True, in some cases the business requires entirely new, custom applications, but it may just as likely need to update existing applications, connect new, modular capabilities to older apps, or pull together SaaS and on-premises assets in various ways to meet changing customer needs.

Cloud-Native as the New Architectural Paradigm

Understanding the architectural context for hybrid IT is essential to grasping the full picture of modernization today. This architectural context bridges cloud best practice and hybrid IT: what we now call *cloud-native architecture*. Cloud-native architecture builds on cloud best practices, taking them beyond the cloud itself to all of enterprise IT.

Cloud-native is even more than an architectural approach. It represents a lens through which we can see the entirety of enterprise IT in a new light. For this reason, I consider it to be a new architectural paradigm.

Today, when you hear 'cloud-native' you're likely to think about Kubernetes, the leading open source container orchestration platform on the market. For now, the best way to get started with cloud-native architecture happens to be implementing Kubernetes – although cloud-native covers the gamut from traditional virtualization to containers to serverless computing.

In fact, cloud-native is even more than an architectural approach. It represents a lens through which we can see the entirety of enterprise IT in a new light. For this reason, I consider it to be a new architectural paradigm.

Cloud-native architecture didn't spring forth fully formed out of nothing, of course. Many architectural trends that came before helped teach us the lessons we needed to learn in order to make cloud-native a reality. These shortcomings of SOA and later, microservices architecture provided the perfect breeding ground for Kubernetes.

In the Kubernetes-fueled cloud native architecture paradigm, we have '*smart endpoints, smart pipes*.' In other words, the microservices give us the ability to put intelligence into endpoints, while Kubernetes' abstracted approach to integration is far more intelligent than simple queuing technology.

Kubernetes' smart approach to integration allows the full dynamic and ephemeral nature of containers to support core enterprise concerns of security, management, and integration – benefits of ESBs in the SOA days, now brought forward to a fully cloud-native architectural paradigm.

What Can You Do Today?

Regardless of how far along your organization is on the road to hybrid IT and cloud-native approaches, there are important tasks that you can accomplish now.

The first goal: understanding the various modernization options open to you, and how to make the appropriate decisions about how to move forward given your particular legacy challenges.

Every organization's situation is different, so it's impossible to offer specific advice in a paper like this one. That being said, the most important general advice is to base decisions on business priorities, not on technology fads or what people consider the ‘latest’ approach to solving a particular problem.

Many IT decision makers have jumped to the conclusion that some legacy asset or another is necessarily bad. As a result, they make poor business decisions to pursue costly and risky replacement strategies that may not even meet the same business needs the original system satisfied.

Other executives make the opposite mistake: they look at the cost and complexity of a modernization proposal, only to reject it and maintain the status quo. Organizations that fall into this trap soon find themselves with technology that limits their ability to compete in an increasingly digital world.

The right strategy, therefore, will likely be a combination of any or all of several modernization approaches:

- Leaving certain legacy assets in place
- Modernizing assets by updating them in place
- Modernizing assets by modularizing them and modernizing certain areas of functionality
- Replacing certain assets as necessary.

Another important consideration is to leverage whatever existing integration technology is already running, as older middleware can continue to serve an important role in a modernization effort.

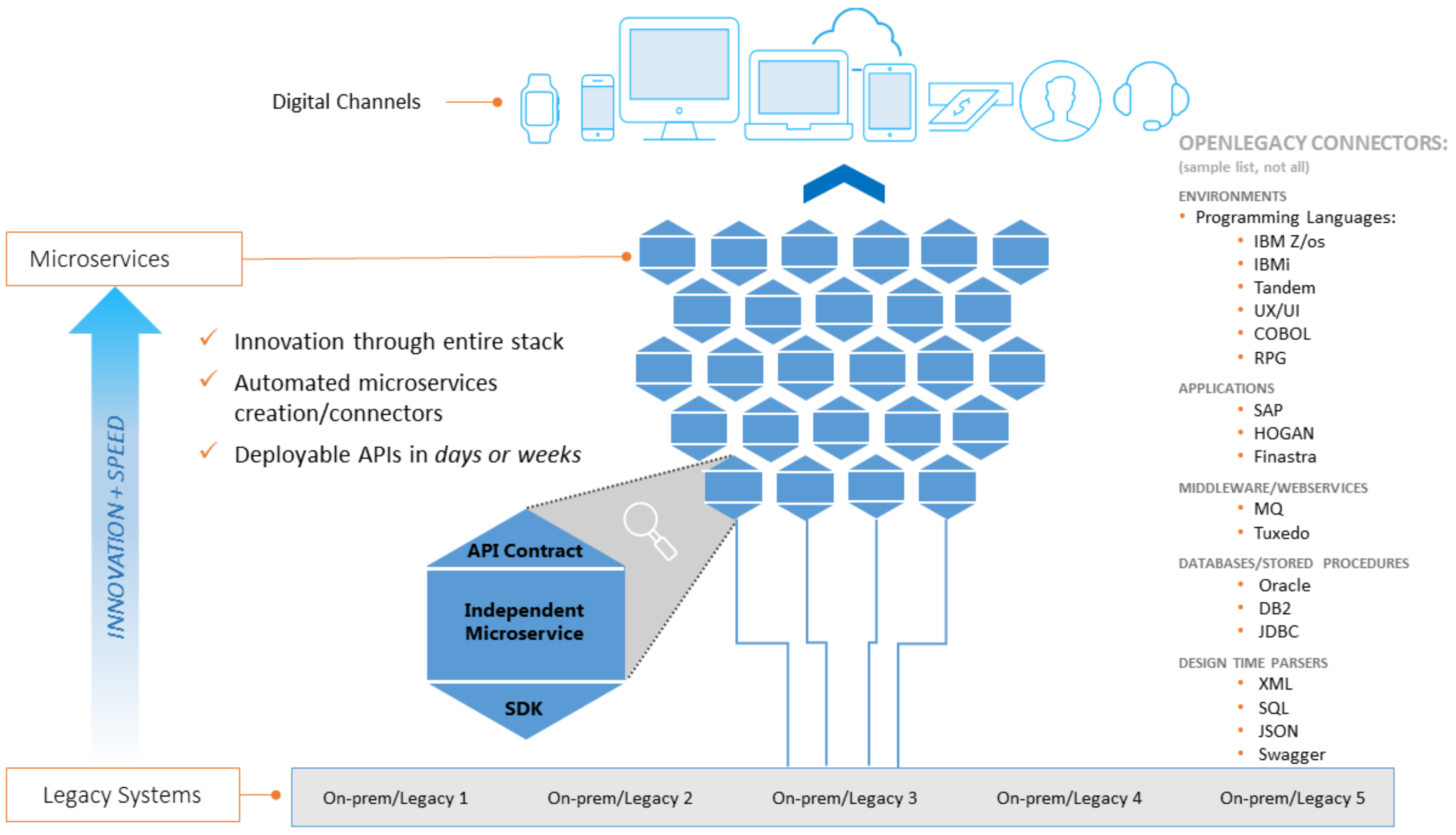
It’s true that ESBs and other pre-cloud technology generally don’t work well with cloud-based assets. To address this problem, OpenLegacy can bypass the middleware altogether, connecting directly to the legacy system. OpenLegacy then automates the process of creating the RESTful microservice API which developers can deploy and access anywhere, even in the cloud.

By bypassing older middleware, this approach yields faster performance than on-premises middleware, because applications connect directly to the legacy system. Furthermore, there is no need for changes to the legacy system itself.

It also makes sense for a modern IT organization to have a cloud-first strategy, where it always looks to the cloud for any purchasing decision, and on-premises investments only take place if no cloud-based option will do.

It's important to emphasize that cloud-first doesn't mean cloud-only – and in the context of legacy modernization, it's more likely to mean taking a cloud-native, hybrid approach to legacy assets than attempting to replace them or migrate them wholesale to the cloud.

Such a cloud-native approach will generally involve implementing microservices within the context of an orchestrated container implementation. Not only can developers create new, bespoke applications with microservices, but with the help of OpenLegacy, microservices can also serve an important role within a cloud-native modernization strategy (see OpenLegacy's approach in the diagram below).



OpenLegacy's Approach (source: OpenLegacy)

There’s no need to think of microservices like some light switch, going from the darkness of legacy suddenly to the light of a fully cloud-native architecture. In reality, modernization is an ongoing process that takes time – and depending on the specifics of the initiative, it can involve different intermediate steps on the way to a fully microservices-based, cloud-native infrastructure.

OpenLegacy understands this reality, enabling organizations to automatically expose legacy assets as SOAP-based Web Services or RESTful APIs, as well as the more modern microservices approach. Which of these technologies is right for you will depend on many factors, including your existing integration infrastructure, skill sets, and the overall progress of your cloud initiatives.

Regardless of where you are on this journey, remember to let your business priorities drive your decision making. OpenLegacy’s approach is ‘digital to the core,’ with microservices-based APIs to meet business priorities, while reducing IT complexity. As a result, OpenLegacy will be with you every step of the way.

About the Author: Jason Bloomberg

Jason Bloomberg is a leading IT industry analyst, author, keynote speaker, and globally recognized expert on multiple disruptive trends in enterprise technology and digital transformation.

He is founder and president of Digital Transformation analyst firm Intellyx. He is ranked #5 on Onalytica's list of top Digital Transformation influencers for 2018 and #15 on Jax's list of top DevOps influencers for 2017, the only person to appear on both lists.

Mr. Bloomberg is the author or coauthor of four books, including *The Agile Architecture Revolution* (Wiley, 2013). His next book, *Low-Code for Dummies*, is due later this year.

About OpenLegacy

OpenLegacy accelerates delivery of innovative digital services from legacy systems in days or weeks versus months. Our microservices-based API integration and management software reduces manual effort by automating API creation, simplifies the process by avoiding layers of complexity, and improves staff efficiency and API performance. Our software directly accesses and extends business logic to web, mobile, our cloud innovations in the form of Java objects, REST APIs or SOAP. Most importantly, this process is not only fast, easy and secure, but also does not require special staff skills or changes to existing systems or architecture. Together, business and IT teams can quickly, easily and securely meet consumer, partner or employee demands for digital services without modernizing or replacing core systems. Learn why leading companies choose OpenLegacy at www.openlegacy.com.