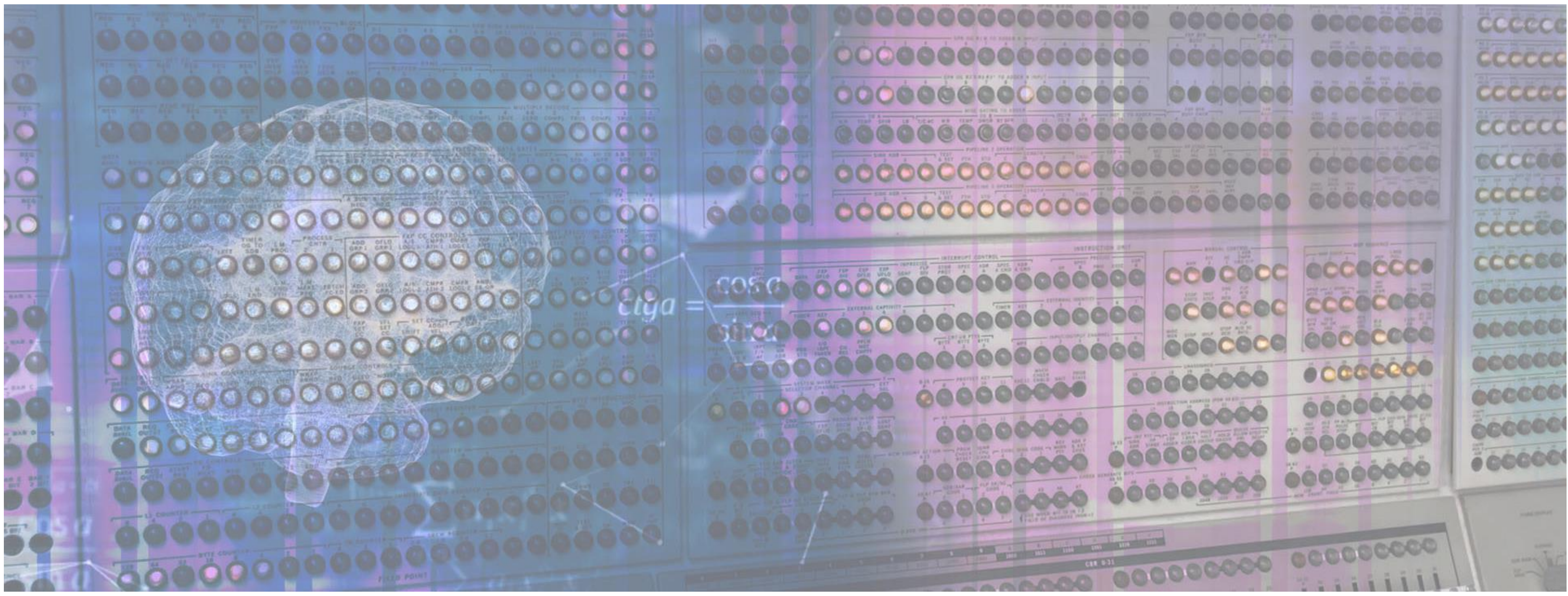


Observing and Managing Integration Infrastructure

Managing Messaging, Event Processing, and Streaming across Hybrid Cloud

An Intellyx Whitepaper for meshIQ
by Jason Bloomberg, Managing Partner
June 2023

Middleware has been an integral part of distributed computing for generations, helping organizations connect disparate applications and software components together.

Nevertheless, middleware has always worked behind the scenes, even though modern applications depend upon it to deliver the performance that business stakeholders require.

Managing such middleware is essential to the proper operation of modern applications, and such management depends upon observability.

Organizations require observability into messages, events, and streaming across their hybrid cloud environments – the MESH that forms the basis of modern integration infrastructure.

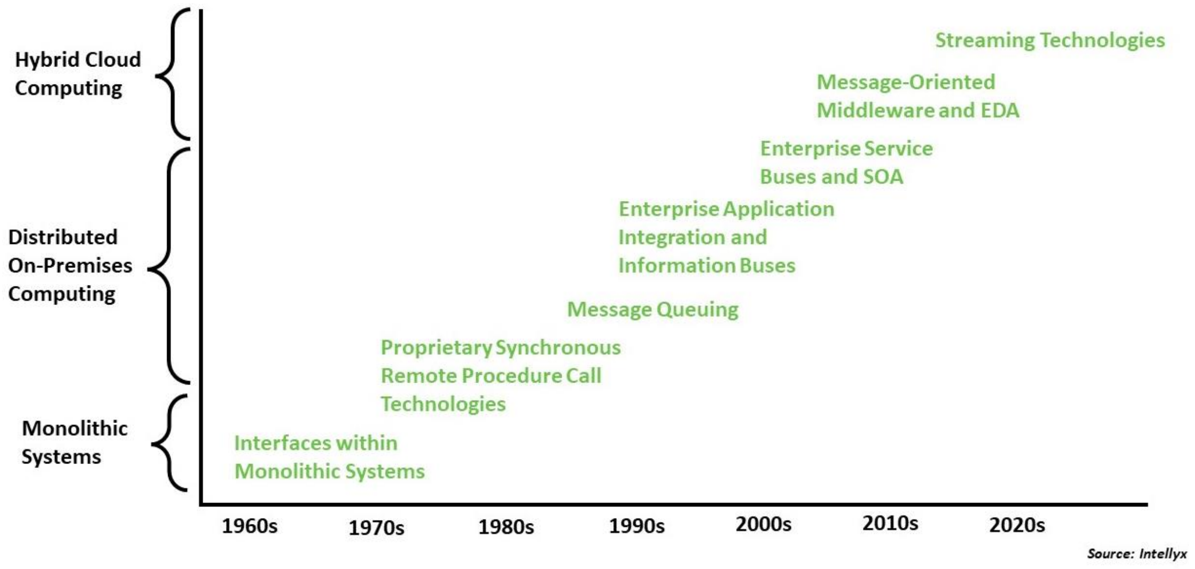
meshIQ provides such observability – into both the middleware and the messages that traverses it, giving organizations the visibility and control they require to ensure their applications perform at their best.

What is Integration Infrastructure?

Modern distributed computing depends upon integration. Organizations scatter their application and data assets far and wide, in various on-premises and cloud-based environments around the globe.

Individual networks and the Internet define the roads that connect these assets, but *middleware* describes the technology that moves information from one system to another – the vehicles on the roads.

The notion of middleware, however, predates the advent of distributed computing by many years, as the following diagram illustrates.


Hybrid Cloud Computing
Distributed On-Premises Computing
Monolithic Systems
Streaming Technologies
Message-Oriented Middleware and EDA
Enterprise Service Buses and SOA
Enterprise Application Integration and Information Buses
Message Queuing
Proprietary Synchronous Remote Procedure Call Technologies
Interfaces within Monolithic Systems
1960s
1970s
1980s
1990s
2000s
2010s
2020s
Source: Intellyx

The History of Middleware

British software entrepreneur Alex d'Agapeyeff coined the term *middleware* to describe the software-based interface between application and system software within monolithic systems at a presentation to a seminal NATO software engineering conference in 1968.

It wasn't until the 1980s, however, that technologists began to use the term middleware to describe the software that connected applications on distributed systems.

Up until the 1980s, stockbrokers, banks, and other participants in capital markets would settle trades using wires connecting the companies, essentially following the nineteenth century telegraph architecture that supported stock ticker machines.

The need for more efficient, cost-effective stock trading and settlement drove innovation across the software industry. These innovations included synchronous remote procedure call (RPC) technology out of Xerox PARC, asynchronous information bus technology out of a precursor to middleware vendor TIBCO, and an early asynchronous message queuing product, IBM MQSeries, which hit the market in 1993.

Message-oriented middleware (MOM), including IBM MQSeries, the TIBCO Information Bus, and others, became the dominant middleware architecture supporting the enterprise application integration (EAI) technologies of the 1990s and the enterprise service bus (ESB) offerings of the 2000s.

Because synchronous middleware like early RPC offerings require a requester to wait for a response before proceeding, such technologies proved too limiting for general distributed use cases where network performance was uncertain.

As a result, message-oriented middleware (MOM), including IBM MQSeries, the TIBCO Information Bus, and others, became the dominant middleware architecture supporting the enterprise application integration (EAI) technologies of the 1990s and the enterprise service bus (ESB) offerings of the 2000s.

Two middleware-centric architectural approaches also matured early in this millennium: ESB-based service-oriented architecture (SOA) and information bus-based event-driven architecture (EDA).

While the practices of SOA and EDA overlapped, they leveraged different integration paradigms. SOA depended on Web Services, which were typically synchronous XML-based software interfaces. In contrast, EDA leveraged the notion of events, where a real-world event (say, a person clicking a button) would kick off a message that represented that event in software.

Today, both TIBCO and IBM have pivoted into other market segments, but their core integration technologies remain in place at many enterprises around the world. In fact, IBM has renamed MQSeries a few times, to IBM WebSphere MQ and now IBM MQ, reflecting its persistence across the integration landscape to this day.

This evergreen nature of a range of middleware products is both a strength and a weakness. They have proven their value through decades of functionality and thus remain a central part of many modern integration architectures.

On the other hand, such older middleware technologies are now legacy, adding more technical debt every year they remain in place. Replacing older middleware, however, is usually not cost-effective. As a result, it remains a maintenance pain point for the organizations that retain it.

Despite the longevity of legacy MOM, the technologies for this type of middleware continue to advance. In addition to the store-and-forward behavior of message queues, more recent advances including publish/subscribe and high-volume streaming technologies have expanded the MOM landscape.

Given the diversity and complexity of applications in the modern hybrid cloud world, most organizations leverage a mix of old and new, synchronous and asynchronous integration patterns, and the middleware that supports them.

Ensuring the proper behavior of modern applications, therefore, depends largely on the middleware that connects their various components. The components themselves may be fully functional, but if they can't communicate with each other properly, then the entire application will grind to a halt.

Ensuring the proper behavior of modern applications depends largely on the middleware that connects their various components. The components themselves may be fully functional, but if they can't communicate with each other properly, then the entire application will grind to a halt.

The Integration MESH Challenge

Managing the availability and performance of modern applications depends both upon the behavior of the applications themselves as well as the middleware that connects them.

There are plenty of products and services on the market today for managing applications, application components like microservices, and the systems that support them. But without the same level of care and attention on the middleware, availability and performance can suffer nevertheless.

Even when application components are fully modern – for example, consisting of containers and microservices in a cloud native application – the middleware that connects the various components across the hybrid landscape is typically a mix of different technologies, both old and new.

To ensure the proper behavior of their applications, therefore, organizations must manage a variety of middleware types:

- *Messaging* – message-oriented middleware is at the heart of most integration. Whether it be some version of IBM MQ or more modern MOM technologies like Rabbit MQ, organizations must have visibility into the behavior of their MOM infrastructure as well as the messages traversing it.
- *Events* – even though we represent events as messages, events differ from other messages in that they can come from any piece of technology at any time. From interactions at user interfaces to alerts from back-end services, EDA covers the lot – and managing such a diversity of events becomes a singular challenge.
- *Streaming* – with MOM and events, something sends a message when something else wants to receive it. Streaming, in contrast, is a steady firehose of data, where consumers of those streams essentially sip at the firehose as needed. When you watch Netflix, you stream one show at a time – but the Netflix infrastructure must stream all the shows all the time. Apply the same idea to corporate data.

Now place these various types of middleware into a *hybrid cloud* environment, consisting of both on-premises and cloud-based assets, perhaps scattered around the world. On-premises integration may still leverage legacy middleware technologies, while the long-distance links require different, more modern middleware and management infrastructure.

Such are the challenges of MESH – *messaging*, *events*, and *streaming* across *hybrid cloud*.

The Role of Observability for MESH

To address these challenges, the first requirement is *observability*. You can't manage what you can't observe. And yet, observing the various middleware and application components isn't sufficient. Organizations must also observe the *topologies*.

If networks are the roads and middleware are the vehicles, then topologies are the street maps. For any distributed application, messages take particular routes depending on process and other business logic.

Just as with vehicle traffic, jams can arise independent of individual vehicles. Understanding the flow of traffic across the map is essential for managing such problems. You need traffic cameras – hopefully on helicopters.

meshIQ provides this necessary level of observability. With meshIQ, operators can track and trace transactions and message flows across disparate application components, providing visibility into the status of messages in flight.

meshIQ provides this necessary level of observability. With meshIQ, operators can track and trace transactions and message flows across disparate application components, providing visibility into the status of messages in flight.

As a result, operators can identify and resolve message bottlenecks quickly, either by addressing the root cause of a slowdown or by rerouting messages temporarily until they can implement a fix.

meshIQ also delivers message statistics, useful for both capacity planning and auditing purposes.

Another challenge that meshIQ addresses is configuration management. With meshIQ, operators can configure, control, and even deploy middleware assets while keeping track of changes across applications and systems.

meshIQ gives operators hands-on visibility and control over individual messages in flight. They can search for messages and then browse, copy, edit, and delete messages as necessary, either individually or in bulk.

meshIQ gives operators hands-on visibility and control over individual messages in flight. They can search for messages and then browse, copy, edit, and delete messages as necessary, either individually or in bulk.

meshIQ then gives operators the ability to test their work in the context of running applications by generating message traffic on the fly.

Typical Industry Use Cases

Even though it remains behind the scenes, middleware is ubiquitous across industries and types of applications.

Retail

In retail for example, the business challenge is maximizing performance during critical seasonal events like Black Friday or other annual sales. As a result, enterprises must implement application architectures across both ecommerce and traditional supply chains with such critical events in mind.

Furthermore, the retail ecommerce customer is particularly sensitive to application performance. Mere milliseconds of response time can make the difference between a sale and an abandoned shopping cart.

For these reasons, performance management, problem detection, and problem remediation must be comprehensive and immediate. Observability and management of the middleware that drives retail interactions must be fully up to the task.

Financial Services

In financial services, regulatory compliance can drive middleware observability and management capabilities. For example, Dodd-Frank compliance (and the corresponding MIFID-II in Europe) requires visibility into trade events.

meshIQ can leverage information in the message payloads themselves, for example, the trade execution time, to provide triggers for service-level agreement alerts.

Insurance

The role middleware plays in the insurance industry centers on the successful completion of processes like claims processing and underwriting.

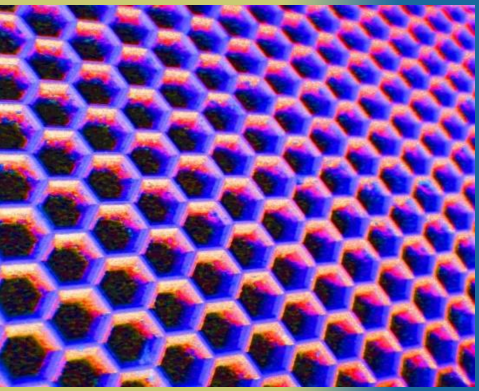
For most insurance carriers, such processes depend upon the interactions of modern and legacy applications – all connected by middleware of various ages. Ensuring the proper functionality of the middleware and thus the successful completion of insurance transactions and other processes depends upon successful MESH observability.

Healthcare

Healthcare also leverages middleware for regulatory compliance, but in the case of HIPAA regulations, compliance constrains interactions at user interfaces as well as medical devices. Furthermore, while compliance is mandatory, it is always secondary to high quality patient care.

meshIQ provides both provider and payer healthcare organizations with the visibility they need into the data flowing through their applications – data whose confidentiality they must rigorously uphold.

Even when the complex regional and global regulatory landscape raises the bar for healthcare transactions, meshIQ can provide the essential visibility into message traffic across diverse, hybrid middleware implementations.



meshIQ combines middleware-specific observability with visibility into the messages and events themselves to discover and visualize the pathways that messages take through the application stack – even when that stack exists in multiple locations across a hybrid cloud deployment.

meshIQ: MESH Observability and Management via a Single Pane of Glass

Every piece of software – middleware included – provides some kind of management interface. Modern technologies may offer standards-based telemetry, while older packages may have API access to management functionality or some kind of proprietary telemetry.

Over the years, meshIQ has implemented its own interfaces to all these management capabilities, beginning with IBM MQSeries, but extending to other middleware including ESBs, MOM, and streaming technologies like Apache Kafka.

meshIQ combines this middleware-specific observability with visibility into the messages and events themselves to discover and visualize the pathways that messages take through the application stack – even when that stack exists in multiple locations across a hybrid cloud deployment.

meshIQ then overlays performance data from each system over the transaction paths that messages take to provide visibility into the entire transaction journey across the application topology.

In this way, meshIQ can leverage AI to identify indicators of anomalies to generate alerts for operators with advice on how to prevent bottlenecks and other middleware issues.

The result is a lowered mean time to recover (MTTR) from bottlenecks and other issues that might occur across the application landscape, extending the mean time between failures (MBTF) that represent the most important key performance indicators for the success of distributed applications in general.

The Intellyx Take

Middleware has never been sexy. Given its essential role as the glue joining other things, its operation is necessarily behind the scenes. Want to see middleware in action? Nothing to see here, move along.

Don’t let that invisibility fool you. The more complex the application, the more important the middleware that holds it together becomes. Given the additional complexities that hybrid cloud environments introduce, it’s a wonder our technology works as well as it does.

Middleware has a decades-long history, but innovation continues apace with the continual improvement of cloud native and streaming technologies. But as is the case with other areas of innovation, enterprises will continue to mix new with old, cutting edge with legacy.

For these reasons, MESH observability from companies like meshIQ is vitally important. Such technology drives business value and mitigates risks like no other technology can, by focusing on the connective tissue that makes modern applications work.

There are also numerous examples of real-time, event-based, streaming apps. Many such applications also leverage distributed environments. One might designate these as GHMAs, but if they are not cloud native, then they fall on the periphery of the definition this paper provides.

Another application type is the fully hybrid application, especially in multi-cloud environments. Enterprises are uncovering various reasons to implement such applications across clouds, either to take advantage of each cloud's specialty or to optimize cloud spend.

An application would fall solidly under the definition of GHMA if it satisfied more than one of these conditions. Perhaps it is real-time and event-based on Kubernetes. Another GHMA might be fully hybrid and event-based. Or perhaps it is fully hybrid while leveraging Kubernetes, perhaps in more than one environment.

Regardless of the specifics of your particular GHMA, running it on Fiorano can give you a leg up on any of these requirements, while taking care of the underlying integration details with its peer servers delivering scalability and high availability – an advantage over building apps using eiPaaS, HIPs, or enterprise low-code platforms could deliver on their own.

About the Author

Jason Bloomberg is the founder and managing partner of enterprise IT industry analysis firm Intellyx. He 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 #13 on the *Top 50 Global Thought Leaders on Cloud Computing 2023* and #10 on the *Top 50 Global Thought Leaders on Mobility 2023*, both by Thinkers 360. He is a leading social amplifier in Onalytica's *Who's Who in Cloud?* for 2022 and a *Top 50 Agile Leaders of 2022* by Team leadersHum.

Mr. Bloomberg is the author or coauthor of five books, including *Low-Code for Dummies*, published in October 2019.

About meshIQ

Headquartered in Plainview, NY, meshIQ provides IT organizations and business executives with the tools and insights they need to understand and manage their digital environments. meshIQ are the global experts at exploiting messaging middleware environments and the contents of messages to improve the business and technical understanding of complex enterprise application stack. Managing Messaging Middleware, Monitoring entire application stacks end-to-end, providing proactive alerts, tracing and tracking transactions, visualizing, analyzing and reporting on machine data in meaningful ways, simplifying root cause analysis, and providing data to support business decisions.