White Paper

Central Role of Messaging Middleware in Cloud and Digital Transformation Initiatives

Sponsored by: IBM

Maureen Fleming
April 2018

EXECUTIVE SUMMARY

Highly decentralized computing is the new normal for most organizations, and digital transformation (DX) initiatives are changing application architectures to event driven to support real-time and near-real-time response cycles. In this environment, enterprises are increasingly turning to messaging middleware to meet the combined requirements of complexity, speed, reliability, and security to connect the digital world of applications and data.

Three major use cases of messaging middleware exemplify this need:

- **Streaming data is used for real-time analytics.** Data streaming from applications, sensors, geolocation services, and social feeds is delivered to analytics systems to rapidly predict and identify problems and opportunities.
- **Microservices communicate with each other via APIs.** When communications between microservices must be reliable, messaging middleware is used in combination with APIs to guarantee delivery.
- **Secure, reliable, and fault-tolerant communications are required when messages are exchanged between applications deployed in different locations.**

Modern messaging middleware portfolios must support diverse workloads, different qualities of service, and a variety of messaging patterns. As data flows within and outside the enterprise to diverse locations (including multiple clouds), end-to-end encryption of messages must be ensured. And messaging software must be portable so that it can colocate with applications to support lower latency requirements. As mission-critical applications move to the cloud, transactionality and fault tolerance are also required.

IBM is the largest provider of messaging middleware in the world. Its MQ messaging family is designed to support the simplest to the most complex use cases. IBM MQ provides native support across diverse public clouds, support for the transport of sensor data, plug-and-play messaging appliances for edge use cases and datacenter consolidation, and scalable messaging suitable for high-volume event-driven data streaming and microbatch delivery.

Technology trends and macrotrends point to the increasing need for messaging middleware over the next several years. To meet these needs, enterprises should consider loosely coupled messaging solutions that support containers for application and middleware portability and provide the ability to apply appropriate qualities of service as needed for individual use cases.

SITUATION OVERVIEW

Dramatic changes in distributed computing, driven by cloud computing and digital transformation, are under way. Consequently, workloads are highly decentralized, yet process cycle times are faster; new types of applications centered around cognitive computing consume an order of magnitude more data than in the past, regardless of where the data originates; and developers must speed up delivery of new functionality to support any location, and any device, across a variety of application types.

Enterprises are learning how to deal with the seemingly contradictory nature of these new requirements:

- How do they speed up cycle times when assets are largely decentralized and solutions must be hybrid?
- How can they deliver the volumes of data at the speeds required to support near-real-time machine learning and cognitive use cases?
- With limited developer resources, how can developers be tasked with delivering new capabilities faster while learning new approaches to building applications and new processes to support development?

If these requirements were isolated to only early adopters, the market impact wouldn't be as significant. But both mainstream and late adopting enterprises are also involved with digital transformation and cloud computing, and the impact of change is felt across functional areas at these organizations as well:

- Today, more than half of all enterprises run at least some workloads in a public cloud. By 2020, over 90% of enterprises will use multiple cloud services and platforms — a transition supported by new investments to manage resources across platforms.
- Today, 75% of enterprises actively involved with public cloud adoption use microservices and APIs to build new applications. Use of microservices will continue to grow as PaaS is deployed on-premises in container platforms. And the adoption rate of containers at large enterprises will exceed 80% within five years from just 7% a year ago.
- More than 80% of enterprise leaders consider digital transformation to be strategically important to their future success.

As a result of these trends, IT executives working with business leaders are trying to figure out how to make it easier to get transformation initiatives up and running consistently. They realize the need to re-architect and replatform around core technology areas that will enable transformation.

Messaging middleware is a critical part of the connectivity layer required for the exchange of data and services across a decentralized architecture, supporting real-time and near-real-time transactional and analytic workloads. Enterprises are looking to messaging middleware to meet the combined requirements of complexity, speed, reliability, and security to connect the digital world of applications and data.

While different messaging middleware products have variations in capabilities, the most common capabilities needed are to:

- Transport data events and service requests from an originating source to targeted endpoints
- Queue, or cache, messages at each target endpoint to ensure messages are received in the order they are sent and persisted until the endpoint processes them or no longer needs them
- Provide qualities of service, such as encryption and guaranteed delivery, to ensure that data is transported reliably and securely

Messaging Middleware in Digital Transformation Initiatives

To illustrate how messaging middleware is used in digital transformation, we use the following set of connected digital transformation use cases and examine each more closely:

- **Messaging in streaming-based predictive analytics.** The marketing department implements a cloud-based streaming solution for near-real-time predictive analytics to determine when to make promotional offers.
- **Adding reliability to stateless microservices.** The development team builds a new microservices-based shopping cart to improve the scalability of the company's ecommerce capabilities.
- **Secure and reliable automation across multiple clouds and the partner ecosystem.** The operations team works with IT to fully automate order processing and fulfillment to ensure orders are processed and fulfilled accurately to meet new two-day delivery service-level agreements.

Messaging in Streaming-Based Predictive Analytics

Messaging middleware is increasingly used in advanced analytics initiatives aimed at predicting problems and opportunities in near real time to gain benefit by responding as quickly as possible.

In this use case, the marketing team wants to analyze social streams, streams of ecommerce log data, and new customer service and new order events. The goal is to predict the likelihood that an existing customer or known prospect will purchase a product or conversely will express negative sentiment about a product or business. Once this is identified, the marketing team wants to determine what the next best action should be, such as making a real-time promotional offer.

The IT team originally wanted to use existing software to deliver data to the analytics, which involved scheduled delivery of bulk data. But it had to shift to messaging middleware because the time window between delivering the new data and evaluating the data to make a prediction was short, and its existing data delivery software was too slow. Other requirements of the IT team included the following:

- The ability to transport from a wide variety of legacy, SaaS, and external data sources both inside and outside the organization
- Scalable messaging to handle high volumes of data
- Delivery of data into subject- or topic-based queues for access by all marketing systems that need access to the data
- Reliable and secure outbound delivery to a service that will automate delivery of real-time offers to appropriate outbound and customer-facing systems

This use case is increasingly common in marketing and across many other initiatives, especially for digital transformation. According to 2017 IDC survey data, roughly 25% of enterprises use predictive analytics to detect and predict problems and opportunities in near real time. For initiatives involving digital transformation, the adoption rate is higher than 50%.

Teams focusing on streaming initiatives realize that messaging middleware is critical to the success of these efforts. According to a different IDC survey covering technology adoption to support IoT analytics initiatives, 50% of the projects involved purchasing new software to transport sensor data and 95% of the respondents are in production or in development using messaging middleware for data streaming. In fact, messaging middleware was the most common technology purchased for initial IoT analytics projects.

For more than 20 years, IT organizations supported analytics workloads by batching and delivering data to a data warehouse or data lake according to a schedule. Organizations did this because it was efficient and cost effective and met the needs of business users who needed the analytics for planning and reporting.

The switch to messaging middleware is accelerating because the choice of technology is no longer about cost but more about how the technology supports the business case. Messaging middleware combined with streaming analytics is justified by the value of rapid, automated insight and rapid responses measured by increases in revenue and decreases in operating costs.

Adding Reliability to Stateless Microservices

New applications are developed differently than they were five years ago. Developers today use microservices, functions as a service, and other event-driven, reactive design patterns to build applications. These applications are constructed as small units of logic, each with its own API, that communicate with each other via events. The use of events allows microservices to operate statelessly.

The deconstruction and the stateless design provide a mechanism to speed up development, inject better reusability, and deliver highly scalable functionality, especially when running on a cloud architecture.

The problem with stateless design is its lack of reliability. Stateless design works well for applications that have limited consequences when an event is not correctly received. An application that allows a user to look up information using a mobile app, for example, is likely to use retry logic when the request from the mobile app to the back-end composite service times out. The consequence is a delay in seeing results on the user interface.

In this use case, the development team is charged with rebuilding a shopping cart using microservices. The developers didn't have any problem building the core stateless functionality of the shopping cart, but the shopping cart itself must be reliable. Without reliability, there are monetary and business reputation consequences when any part of the shopping cart fails. Reliability also extends to messaging-based interactions with the third-party payment service, inventory systems, and other applications used with the shopping cart.

Reliability is stateful. If a microservice has to embed reliability, it really isn't stateless anymore and many of the benefits of stateless design are not fully present. In this use case, and in many other use cases, developers have had to find a way to create reliability without breaking the stateless paradigm of the microservice. Messaging middleware is an important mechanism for doing this.

Messaging middleware is increasingly used for transmitting and queuing events. Event queues are effectively caches that temporarily persist the event until it is processed. Messaging middleware also has guaranteed delivery mechanisms that communicate with the source of the event to acknowledge receipt. In addition, messaging middleware must execute once-and-only-once logic to dedupe events when the resend logic delivers the same event more than once.

In a message queue–based approach, stateless services execute separately from the queues and guaranteed delivery logic, providing a means to maintain the stateless nature of the business logic without any reliability overhead, while the messaging middleware provides the reliability.

The relationship between events, business logic, and guaranteed delivery has been around for a long time in systems that are distributed and execute asynchronously, particularly for mission-critical transaction use cases. Messaging middleware began to fall out of favor with the newer styles of developing applications because messaging became more dependent on centralized control.

Development teams responded by shifting to API-based approaches, which were usually implemented with API gateways. But the complexity of building and managing reliability for each microservice, much less the need to create some type of queuing mechanism, made teams realize messaging had a role in this new architecture. Vendors began to respond by re-architecting their messaging to begin offering capabilities that shifted routing and reliability logic to the endpoints.

Secure and Reliable Automation Across Multiple Clouds and the Partner Ecosystem

As enterprises focus on systematically increasing their levels of automation, the combination of APIs, process orchestration, and secure and reliable messaging plays a central role. In this use case, an enterprise has to fully automate all processes touching the shopping cart order to comply with a new two-day delivery service level to better compete in the market.

Once an order is placed, it must be processed by the organization's financial systems, updated in the SaaS marketing application, delivered to the organization's data warehouse and to the new cloud-based streaming analytics solution, and either shipped directly from a manufacturer or fulfilled by warehouse and logistics partners.

The company previously delivered orders by batch using the EDI x12 format standard to manufacturers and logistics providers. Because the cycle time changed, it shifted to an API-based approach using secure and reliable messaging. Fault tolerance was also required to ensure none of the systems lost any part of the transaction. Process orchestration was used with the messaging to keep track of process completion acknowledgements to ensure that each step was executed accurately.

Messaging middleware was containerized and deployed locally with each application in the datacenter, colocated with each cloud involved with the orchestration and with partners to improve speed and reliability.

Messaging middleware also provided:

- Messaging transport to deliver instructions to the correct endpoints
- Message reliability for processes involved with complex business transactions that need that quality of service
- The ability to execute a complex transaction, change the transaction, or back out of the transaction through compensation — by undoing the entire transaction or parts of the order when the order is canceled or changed
- End-to-end encryption to protect the privacy of each message
- Fault tolerance to recover from outages

Today, there is also a push to containerize messaging to colocate with microservices running in a container platform. IDC surveys indicate that containers will be adopted rapidly by enterprises, passing mainstream adoption in less than five years. This shows the pent-up demand for portability of both workloads and utility-like services across cloud, datacenter, and edge locations.

In this era of digital transformation, enterprises widely recognize the need for a frictionless customer experience. And that requires an understanding of how a transaction or a process must be executed, as well as reliable communications to all systems responsible and situational awareness of where the order is in its process.

THE NEED FOR COMPLETE MESSAGING MIDDLEWARE

The three use cases illustrate the need for:

- A mechanism to stream data as part of a solution to predict when a prospect is ready to receive a promotional offer
- Reliability to ensure an application works consistently every time a user or system is using the application or service
- Transport across decentralized systems, including persisting the information and coordinating reliable execution as part of a transaction
- Security to ensure that the data contained in a message is fully protected from source to destination
- Fault tolerance to ensure that no message is lost during transport or in a queue

The core value of messaging middleware is its ability to transport messages securely from any source to any subscribing endpoint and do so reliably and at scale. At the same time, there is a need to handle common problems, such as reversing or compensating order changes or cancellations and deduping events when there are retransmissions.

Not all use cases require the full set of capabilities, which means messaging middleware that is loosely coupled and able to provide only the qualities required for each use case is important.

For organizations replatforming or extending messaging middleware, gaining the full benefit of the investment will come from adopting a messaging middleware solution that is engineered to provide the capabilities described previously, in addition to the following capabilities:

- The ability to recognize and translate across diverse messaging protocols
- Delivery of both data and files
- Self-service access to messaging services through messaging software's APIs
- Out-of-the-box connectivity to common SaaS application APIs and interoperability with legacy environments
- Deployable across multiple clouds (either natively or on container platforms) and edge locations, providing much-needed messaging application embeddability and portability
- Interoperability with other technology used to perform similar functions
- End-to-end encryption
- High-availability options across all deployment locations and disaster recovery features for out-of-region recovery
- Centralized administration and management of messaging deployed across all locations

IBM MESSAGING MIDDLEWARE

IBM is the largest provider of messaging middleware worldwide. Its roots are in providing message queuing for mainframes and legacy systems, and these messaging capabilities have evolved through successive generations of technology and architectural changes, providing the messaging layer used to support critical business systems in many of the world's largest companies.

Today, IBM messaging middleware is offered on a variety of platforms including cloud architecture, virtual machines (VMs), and containers:

- **IBM MQ** provides high-speed, end-to-end encrypted, once-and-once-only messaging, supporting both transactional and data streaming use cases, as well as messaging and queuing for communications across applications and among the microservices of an application. MQ runs natively on the following clouds: IBM, AWS, Azure, and Google Cloud Platform. MQ supports Docker and is deployable into Kubernetes-based container platforms. And it runs on-premises on Windows and Linux and on a variety of legacy environments including natively on z/OS. MQ also supports a range of high-availability and disaster recovery options. Specifically, MQ's features include:
  - Once-and-once-only delivery of messages
  - Transactional units of work (XA)
  - End-to-end encryption with no application changes and no performance impact
  - Horizontal and vertical scaling
  - High availability and disaster recovery across all environments
- The **MQ Appliance** is a high-specification physical appliance deployed in datacenters or remote locations or in areas without local skills and where high availability and disaster recovery are important. The MQ Appliance provides a locked-down, highly available, and secure messaging environment. It is preoptimized for simplicity and is designed to be simple to configure, deploy, and maintain.
- **MessageSight** is a container-based solution for machine-to-machine and mobile communications supporting the MQTT protocol, which is also supported by IBM MQ in the MQ Advanced offering. MQTT can be used to connect from outside the enterprise into MessageSight, connecting to applications inside the enterprise.
- **Message Hub** is Apache Kafka as a service on the IBM Cloud, supporting high-volume data streaming.

A variety of tools designed for the configuration, management, and monitoring of MQ are provided with the product, including a browser-based tool, an installable Eclipse tool, APIs, and command-line scripting. Most customers use a combination of these tools in addition to external vendor tooling, which exposes system-level MQ status into a wider system management tool.

CHALLENGES/OPPORTUNITIES

Key challenges include:

- **Determining when to shift data delivery from batch to streaming.** Organizations typically adopt technology for known requirements and try to future proof their selection as much as possible. It is impossible to future proof the shift to event-driven design with technology that works in batch. Even two years ago, our discussions with clients centered around reluctance in shifting to a near-real-time architecture. Today, many clients assume the shift is required and are adopting messaging middleware to support event-driven use cases.
  - **Recommendation:** For teams working in enterprises investing in digital transformation, assume cycle times will become significantly faster. Assume development will focus on an event-driven architecture, and purchase software platforms and tools accordingly.
- **Designing short-term persistence and reliability into a microservices architecture.** Advocates of microservices tout the benefits of stateless design and communications solely by calling the APIs of other composite services and microservices. Developers designing solutions that require reliability are not sure they should build the solution on microservices. Thinking has evolved, and there is a much greater understanding that both reliability and state are attributes that have to be supported, even when the business logic of the applications is constructed using microservices.
  - **Recommendation:** There are different ways to implement state to support microservices, but messaging is unique in its ability to provide reliability. Messaging and message queues should be used when financial and reputation risks are the consequences of unreliability.
- **Automating fault-tolerant, secure, and reliable hybrid processes.** Today, nearly half of the enterprises that have adopted cloud for SaaS applications or custom applications operate under a cloud-first mandate. That mandate extends from new development to replatforming. Enterprises continue to struggle to build reliability and fault tolerance into automated mission-critical processes that cross domains, including clouds and partner ecosystems.
  - **Recommendation:** There are wide variations in messaging middleware product support of qualities of service, especially on a public cloud. Once-and-only-once logic and fault tolerance are just two examples. Full encryption and the ability to embed into containers are also differences among messaging solutions. Each of these features is required to support reliable hybrid applications and processes. When doing a proof of concept to evaluate messaging middleware, make sure these capabilities are tested.

Key opportunities include:

- **Aligning messaging middleware with an enterprise's digital transformation initiatives.** Often, the development team working on DX is different from the development team maintaining existing enterprise systems. But successful DX is dependent on leveraging an enterprise's existing processes and assets. Replatforming and change efforts should involve making the enterprise's existing capabilities DX compatible and DX ready, extending with new capabilities as needed. While messaging is ubiquitous in most large organizations, existing messaging middleware products often have to be extended or replatformed to support cloud and hybrid use cases.
  - **Recommendation:** Identify new cloud projects that need to communicate with existing applications, and evaluate existing messaging to identify gaps in functionality that would create problems. Gaps in security, cloud messaging high availability, microservices reliability, and overall reliability may be present and will create an opportunity to adopt new messaging middleware to bring existing on-premises qualities of service to hybrid use cases.

- **Replatforming messaging middleware to work in containers.** Complex applications and utility-grade services that eventually need to run on a public cloud are often replatformed and deployed in stages. Container platforms that are hosted or run in an enterprise datacenter are a new approach to replatforming that is compatible with a staged approach. Replatforming to a cloud architecture using containers running on existing infrastructure is the first stage. Messaging middleware is often a core part of the application or the utility service, and embedding messaging in the container platform may be required. Enterprises are also looking at containers as an enabler of portability, moving workloads where they can be optimized.
    - **Recommendations:** Container adoption is unfolding at a substantially faster rate than cloud adoption. Any new investment in messaging middleware should be for products that are already available as a Docker image.

CONCLUSION

Enterprises will increasingly need messaging middleware over the next several years as applications are deployed across multiple clouds and across an enterprise and its ecosystem of partners. This application sprawl will drive a greater dependence on applications being able to communicate with each other securely and reliably over distances.

High-end messaging portfolios have significantly evolved from the monolithic messaging offerings from a decade ago. They have gradually evolved to become portable and loosely coupled, with configurable capabilities and qualities of service that allow enterprises to deploy lightweight and embeddable messaging to achieve the scale required by today's demanding use cases.

About IDC

International Data Corporation (IDC) is the premier global provider of market intelligence, advisory services, and events for the information technology, telecommunications and consumer technology markets. IDC helps IT professionals, business executives, and the investment community make fact-based decisions on technology purchases and business strategy. More than 1,100 IDC analysts provide global, regional, and local expertise on technology and industry opportunities and trends in over 110 countries worldwide. For 50 years, IDC has provided strategic insights to help our clients achieve their key business objectives. IDC is a subsidiary of IDG, the world's leading technology media, research, and events company.

Global Headquarters

5 Speen Street
Framingham, MA 01701
USA
508.872.8200
Twitter: @IDC
idc-community.com
www.idc.com