INFOVIEW
CONNECT, REVITALIZE, SUSTAIN

Unlock IBM i (AS/400) data and processes

via API-led connectivity

Integrating IBM i systems

There is no question that one of the biggest challenges of modern IT is a massive IT delivery gap due to disruptive market forces and growing business demands.   Technology is changing at a rapid pace, allowing businesses to increase productivity, reach new markets in creative ways and provide a greater level of customer service, all of which increase the bottom line.

> Companies that are running on IBM i (AS/400) based systems are in a unique position to take advantage by implementing a well architected and executed integration strategy.

This paper reviews the primary drivers for integration, the challenges faced by CIOs trying to meet the changing business requirements by leveraging their company's investment in the IBM i (AS/400) based systems, and the successful integration story of a company that utilized the API led approach to unlocking the back-end data and business logic.

Primary drivers for integration

- **Leverage existing business logic and processes**

Expose IBM i data and processes as reusable assets / APIs and consume the APIs provided by other systems, partners, vendors and customers.

- **Seamless integration of applications and data**

Provide a seamless integration of applications, data and devices by using a common technology and engagement infrastructure for publishing, consuming, discovering, and collaborating on the reusable data and process APIs.

- **Self-serve IT**

A new operational model has emerged, shifting the Central IT away from its traditional role as a sole technology provider. IT builds, operates and governs core data and processes via the System APIs. Business, partners, vendors and customers consume the APIs to rapidly deliver their own projects.

Challenges faced by CIOs

> While the foundation is solid, the applications that many companies are running on IBM i have been developed many years ago and present a number of challenges.

The IBM i is a modern, secure, stable and highly sophisticated operating environment that requires little or no management. It is designed to securely and predictably run diverse workloads such as core line of business applications, focusing on quality of service and availability. The system claims one of the lowest Total Cost of Ownership for integrated on premise solutions.

These are all valid reasons why you should consider integrating your IBM I with your other systems. There are, however, several challenges that must be considered and dealt with:

- **Responsiveness to business requirements**

IT traditionally focuses on incremental changes and maintaining legacy applications, often becoming a bottleneck of Digital Transformation initiatives.   Businesses must rapidly deliver new products and services while keeping the efficient operational models and offering customers multiple channels to interact in real time.

- **Closed proprietary nature of applications**

Data and processes running on IBM i based monolithic applications are traditionally slow and expensive to unlock

- **Technical debt**

Large monolithic applications have been developed over the course of many years by various teams with implicit dependencies between application modules, often inefficient and redundant code. Changes are risky and take a long time to implement. Issues are hard to troubleshoot.

- **Point to point**

Piecemeal integration strategies using a variety of point to point technologies, typically file-based batch processes. The systems become tightly coupled, the interfaces are brittle, hard to manage, slow and expensive to change and reuse.

- **Data issues**

The Integration team has to understand the database model, implicit rules around lists of values etc. In many cases the data must be “scrubbed” before it can be used by outside components. With a direct DB interface, the external processes may accidentally impact DB engine performance by running unoptimized queries.

- **Green screen user interface**

A text based UI, often with embedded business logic, slows the onboarding of new system users, limits reuse via modern web and mobile clients

- **Skill gap of IBM i team**

Typically, the IBM i team doesn’t have extensive integration experience, and applies AS400 application development design patterns to integrations, resulting in a point to point spaghetti bowl

- **Expensive tools**

From IBM and modernization vendors

The opportunity – API-led Connectivity

The solution to solving these challenges is API-led connectivity.

> The fundamental building blocks of an API-led architecture are purpose driven development of API’s in order to meet application requirements, while establishing policies and managing access to backend data.

In most of the integration scenarios IBM i plays a role of System API provider or Experience API consumer. The IBM i supports a wide variety of built-in and commercially available integration tools and options to implement both of these scenarios. With the choice comes the challenge of selecting a right tool.

Below are some considerations for selecting a right local architecture, design and a set of tools:

- **Consistency**

All use cases that fall into the same category should be implemented using the same architecture, design and set of tools. For example, IBM i Integrated Web Services server (IWS) is a quick and relatively easy tool to expose simple RPG programs as REST or SOAP web services and could support a number of integration use cases. However, the IWS has some limitations on passing the request and response data, performance, security, as well as introduces another runtime (Apache server with IWS). For transactions with large complex payloads or high transaction volumes, such as sales order, the alternative approach or tool would be required.

- **Learning curve**

Reuse is important for rapid service delivery. This applies to both application assets and the existing skills. Teams should work with the technologies and tools they are already comfortable with. IBM i development teams can naturally work with DB2 database tables and call IBM i programs from their integration layer. By contrast, it requires a significantly more effort (time and risks) or cross-functional developers with prior experience to parse XML, call HTTP endpoints, execute Java methods from RPG.

- **Operational data insight**

The IBM i integration toolset should capture key operational data for all inbound and outbound transactions. The data can then be used for ad-hoc monitoring and alerting, or streamed into the Business Activity Monitoring tools such as Splunk, that can implement proactive alerting based on transaction processing time as compared to historical averages etc.

- **Scalability**

The IBM i integration architecture must be able to scale with increased demand, and ensure the service availability while limiting the system impact from the peak volumes. Typically, this is achieved by using Data Queues and asynchronous processors for communicating with external systems.

How to achieve API-led integration?

- **Access IBM i business data**

From the consumer point of view, the data is provided via the API. The data access can be implemented via direct database queries and / or by executing the IBM i data extract programs. For example, product name and features can be retrieved directly from the merchandize management system's database, while the product pricing information would most likely be returned by a program that contains the business logic of applying various pricing catalogs and discounts based on the customer, product, location, date, promotion code, and other parameters. For consistent integrations, the recommendation is to always serve IBM i business data via API.

- **Execute IBM i business function**

Consumers call the API that in turn calls an IBM i program and facilitates request / response data exchange in the case of complex data structures. For example, when passing a sales order that includes a list of products, payment methods, fulfillment methods and other information, it does not easily fit into program parameters. Typical API implementation for this use case is to stage the data in the database then call IBM i program to process the data.

- **Stream IBM i business events**

In this case, back-end function publishes the event (such as order status change or customer request) via a special API, playing the role of service consumer. It is common to send the events asynchronously to minimize system impact, balance the load and improve reliability. Depending on the nature of the event it may only contain the ID and new state of the entity, or the entire entity (such as customer details) can be attached to the event and sent for consumption. Integration platform routes the event to subscriber(s) applications and publishes using their APIs.

- **Access external data from IBM i processes**

In this scenario, the IBM i application consumes the external API. There are various techniques and methods to access remote databases directly from IBM i program (for example using Java / JDBC wrappers) however this approach is not recommended due to downsides of direct point to point interfaces. A better option is to publish the API that implements data access to remote systems into the integration platform, then IBM i process can consume the API to get the data.

- **Execute external business function from IBM i**

The back-end program will consume an API exposed by the integration platform. There's a number of tools that aim at insulating back-end development teams from any knowledge of API calls.

- **Subscribe to the external business events on IBM i**

The IBM i application provides an API to receive the event notification from the integration platform. It is common to queue the incoming events and process them asynchronously to improve the throughput and minimize system impact due to volume spikes.

Integration in action

Mulesoft Anypoint platform facilitates the API led connectivity approach, application network, connects anything to anything, translates between XML, JSON, IBM i database structures etc.

The AS400 connector simplifies integration with IBM i systems via data queues and command calls

Infoview Web Transaction Framework streamlines and accelerates the AS/400 integration layer development, hides complexities of dealing with data queues and external systems, provides a lightweight and straightforward development and operational model for IBM i integration programs

Together the Mulesoft Anypoint platform, the AS/400 connector and Web Transaction Framework provide an end to end solution for rapidly unlocking IBM i data and processes, and consistently enabling IBM i applications to consume external APIs.

- **Case Study: Unlock legacy ERP data and business processes**

A Specialty Retailer of durable consumer goods and related services, operating over 100 stores, warehouses and cross-docks, and rapidly expanding into new geographic areas needed to modernize their legacy ERP system by migrating the non-differentiating components (such as Warehouse Management, Delivery, Service, POS and other supporting systems) to industry standard solutions while streamlining the code base for the core applications.

The Challenge

Their home grown ERP system, running on an IBM Power Systems i (AS/400) was developed over the course of 20+ years and had several shortcomings that limited IT and Business agility:

- Difficult to share application data and reuse business logic: The integrations with other components, partners and vendors were implemented as point to point interfaces, typically file based, using a variety of disparate technologies and tools.

- Slow and expensive to change and maintain: Over the years, the system codebase accumulated high levels of technical debt.

- Hard to migrate a particular system / component to a new platform: The application modules are tightly coupled via shared database and direct program calls, resulting in large number of implicit dependencies.

As a result, the IT staff spent much of their time putting out fires and keeping the lights on, as opposed to innovating and enabling business growth.

The Solution

The Specialty Retailer partnered with Infoview Systems to implement the Mulesoft Anypoint platform as a standard integration environment. They selected Mulesoft based on ease of use, lightweight architecture, high performance, large number of out of the box connectors, seamless hybrid cloud / on-prem deployment model and subscription licensing. The Retailer decided to start with the on-premise Mulesoft runtime that offered more options for solution governance and security. The Retailer leveraged Infoview System's expertise in architecting and delivering complex integration and legacy modernization solutions. In addition to the core strengths of the Mulesoft Anypoint platform, the Retailer utilized the IBM AS/400 connector for Anypoint Platform and AS/400 Web Transaction Framework products developed by Infoview Systems to accelerate the delivery of their integration projects.

All three interfaces were ready for testing within a few weeks and went live in less than 5 months with minimal post-production issues.

Benefits

The Mulesoft Anypoint platform, coupled with the AS/400 connector and Web Transaction Framework was the correct tool set for unlocking the legacy ERP data and business logic, and connecting it to other enterprise systems, vendors and partners.

Infoview System's architects and developers with cross-platform AS/400 and Mulesoft skill sets were instrumental in the successful implementation and ongoing development.

When connecting systems, the team should avoid a point to point mentality and instead follow the API led connectivity approach (System / Process / Experience APIs) even if customers don't yet have an API management platform.

Connecting, modernizing or migrating off of legacy systems is a complex undertaking. Success rates are much better with a capable cross-functional delivery team with both Integration and IBM i experience. **To learn more about our professional services, MuleSoft's Anypoint Platform, the AS/400 connector and Web Transaction Framework visit www.infoviewsystems.com**