WHITE PAPER

B2B MIDDLEWARE MODERNIZATION

A Program Management Perspective

Abstract

This white paper provides a program management perspective on the modernization of B2B (business-to-business) integration middleware to a new platform for a large organization. The paper covers various aspects of B2B modernization such as key challenges from a people, process, and technology perspective, the modernization framework and approach, and the success drivers for modernization.

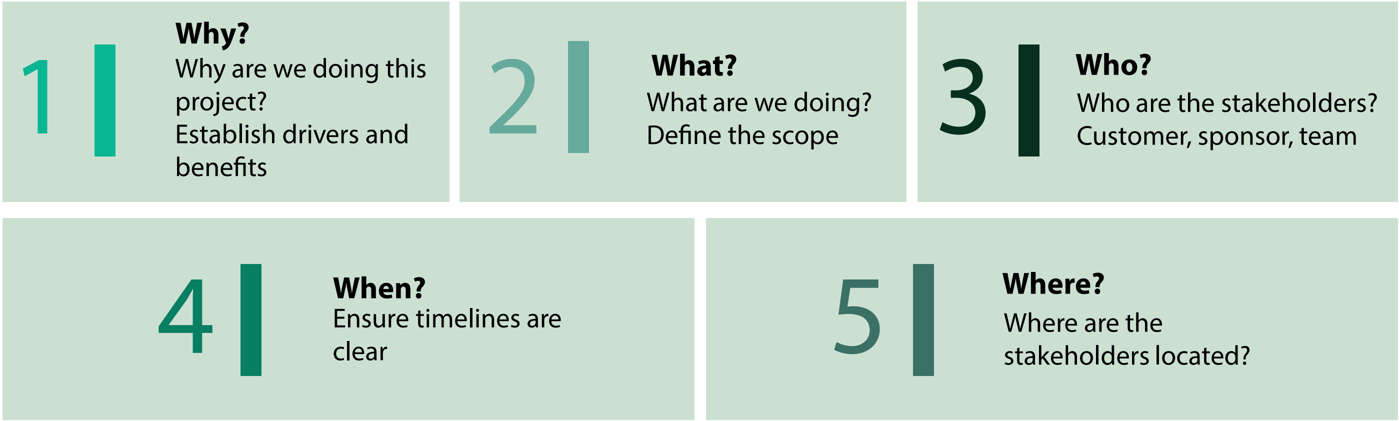
Infosys®
Navigate your next

Table of Contents

1. Before we begin

Middleware modernization must be aligned with an organization's IT strategy. Assess the new platform based on how well it is able to support the organization's business goals. For a B2B organization, the middleware must enable appropriate management of trading partner connections, secure transmission of data, auditable deletion of expired data, workflows to escalate system errors to users, and business user visibility of the end-to-end flow of data that spans business processes. The focus should be on enabling users to accomplish all tasks in an easy-to-manage, easy-to-learn, and a user-friendly interface.

Before project execution begins, it is important to make sure there are clear answers to some fundamental questions as shown in Figure 1. With the right answers, it is easier to frame an approach to plan and execute a program of this scale.



1 **Why?** Why are we doing this project? Establish drivers and benefits

2 **What?** What are we doing? Define the scope

3 **Who?** Who are the stakeholders? Customer, sponsor, team

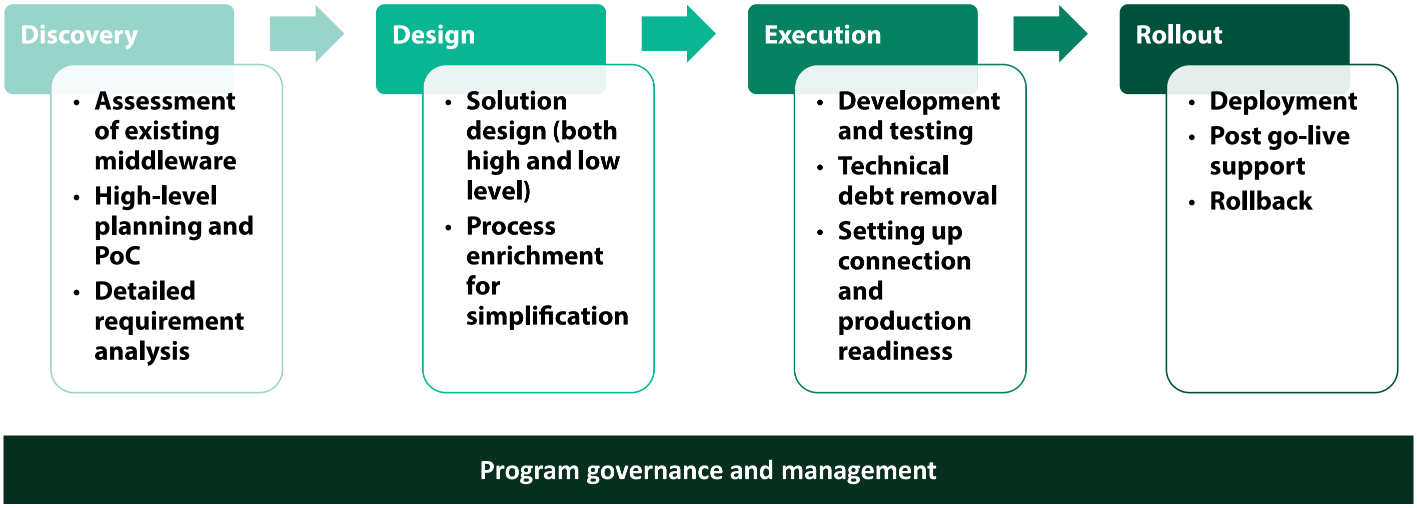
4 **When?** Ensure timelines are clear

5 **Where?** Where are the stakeholders located?

Figure 1 – Fundamental questions to answer before beginning project execution

2. Framework

A successful middleware modernization project has four clearly defined phases of execution as outlined in Figure 2.



Discovery
- **Assessment of existing middleware**
- **High-level planning and PoC**
- **Detailed requirement analysis**

Design
- **Solution design (both high and low level)**
- **Process enrichment for simplification**

Execution
- **Development and testing**
- **Technical debt removal**
- **Setting up connection and production readiness**

Rollout
- **Deployment**
- **Post go-live support**
- **Rollback**

Program governance and management

Figure 2 – Program governance framework

3. Phases in middleware modernization

As shown in Figure 2, there are distinct phases in executing a middleware modernization project. This section describes the various steps involved in detail.

a. Assessment of existing middleware

In the discovery phase of the modernization project, it is important to understand the complexities of the middleware involved to architect the right solution.

Identifying existing interfaces

1. The first step is to identify all interfaces, their end-point systems, and communication protocols. Analyzing the existing middleware interface report (MIR) is an important part of this assessment
2. The next step is a high-level study of the mapping logic, configurational rules, data flow and its volumetrics, and usage of custom code and scripts for each interface involved
3. The final step is to understand the non-functional requirements as they become the base for the new system. For example, we should consider information about existing SLAs, in-built reports, role hierarchy, governance and resource profiles

Grouping interfaces based on patterns

After identifying all interfaces, group them based on technical patterns identified. Grouping of interfaces will divide the whole modernization into small pieces. Common patterns based on which interface grouping is done are:

- **End point connections:** All interfaces with same source and target end-points
- **Functionality:** All interfaces related to a specific functionality
- **Geography:** All interfaces related to a geography
- **Criticality:** Categorizing all interfaces into high, medium, or low based on the processes they support
- **Business processes:** All interfaces that support specific business processes

b. High-level planning

At a high level, here are some key aspects to consider while defining the overall plan for modernization.

Prioritizing interface groups

After grouping interfaces based on identified patterns, assign a priority to each group. Prioritization is based on criticality and complexity of interface groups as defined by the business.

Conducting PoCs

Schedule proof-of-concept (PoC) exercises in the initial stage of the project to ensure you are on the right path. This may require building a process template or framework specific to each interface group. You can continue to enhance the PoC while executing the complete modernization.

Multiple PoCs are required to evaluate where processes can be automated to reduce manual effort and provide better value to the customer. It is critical to collaborate with in-house center of excellence (CoE) teams to leverage accelerators and generate additional value for clients.

Stopping license renewal for legacy systems

Companies do not want to pay double license costs - for their legacy system and the new system - during the modernization journey. Proper planning is critical to ensure that the modernization is complete within the license renewal window for the legacy system.

c. Detailed requirement analysis

A subject matter expert (SME) must analyze each interface based on parameters described in Table 1.

Parameter	Description
Mapping logic	Analyze and understand all transformations running in the legacy system middleware as the same transformation must be mapped in the new system. Conduct a thorough assessment of transformation logic, custom code or scripts and configurations to ensure custom requirements are executed accurately during the modernization.
Data dictionaries	Analyze all libraries, lookup values, and mapping tables that are part of the legacy system for enrichment, simplification, and rationalization. This exercise will also help in reducing the technical debt.
End points	Document all end point connections such as source and target systems and integration patterns between them such as scheduled vs event-based, web services vs messaging, synchronous vs asynchronous to avoid ambiguity during migration.
Message formats and authentication scheme	Note down message formats for each message type and their authentication such as key-based, password-based, PGP encryption or any other type of encryption.
Complexity, criticality, and amount of work	Evaluate each interface in terms of complexity of logic built, criticality to business operations, and amount of work involved. These parameters will help the delivery team prioritize the interfaces and plan the migration accordingly.

Table 1 – Parameters used to evaluate interfaces for modernization

d. Designing the new middleware architecture

Analyze the existing interfaces and prepare a high-level architecture for the to-be state. Typically, middleware modernization involves rationalization and simplification of other interdependent and connected legacy systems.

Assess infrastructure requirements

While revamping the middleware design and architecture, it is important to assess the infrastructure requirements. Identify the infrastructure requirements in the initial phase and highlight challenges early to avoid any risk to the overall execution.

Baseline application performance

Before starting a migration project, it is important to baseline the actual application performance with expected performance. To define this baseline, assess the current product speed in terms of processing rate, response times, and resource usage (CPU utilization and memory consumption). Your assessment must also consider message flow statistics such as hourly volume, peak volume in both directions (inbound and outbound), and maximum file size.

Design controlled message flow

Controlled message flow to avoid overload on the system is one of the key aspects while designing middleware architecture and is done through throttling. Important factors to consider for throttling are:

- **Maximum concurrent processing:** To restrict the number of messages that can be concurrently processed by the middleware
- **Queue length:** Throttling queue length defines the maximum number of messages that can be held in the queue at any point of time. Once the queue is full, new messages are discarded
- **Message expiration:** The period of time for which a message can stay in the queue

e. Development and regression testing

The delivery team can start building the system based on the prioritization defined in agreement with the business team after the interfaces have been analyzed and the design is in place. It is best that the support team reviews each interface built to ensure robustness of design and implementation.

Building transformation logic and processes

While building processes, leverage templates, accelerators, and process frameworks from the CoE. Enhance the process templates to incorporate non-functional requirements such as error handling, message archiving, process monitoring, and document tracking as required by business users.

Testing interfaces and processes

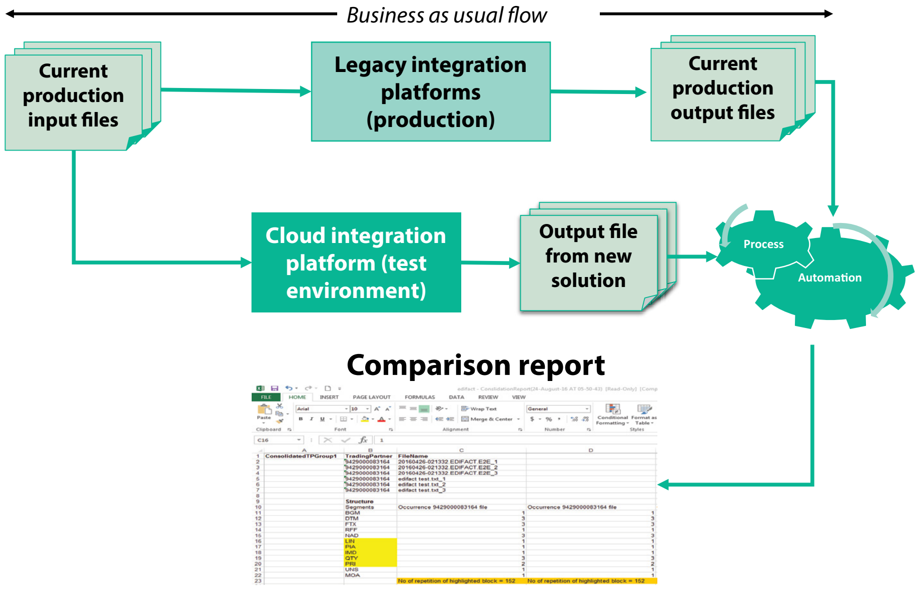
Test the transformation logic of each interface by extracting messages from the legacy system and running these through the new system. Compare the outputs of the new and legacy system for the same input message.

The number of messages with which testing should be done for an interface depends on the volume of messages through that interface, complexity, and criticality of that interface from a business perspective. The best practice is to run all messages from the last three months along with all quarter-end and year-end messages through an interface to validate the transformation logic.

Automation testing

To cover all business scenarios, it is important to test as many files as possible. But this might run into millions if we consider few thousand files per interface. Manually testing millions of files is an impossible task.

Figure 3 depicts one of the processes that can be leveraged for bulk data comparison and to reduce manual effort, ensuring quick delivery and improved accuracy. This process can run daily for a defined time-period (for example, 3 months) on a regression testing tool so that there is maximum coverage of the type of files exchanged with the trading partners. The comparison report provides insight into the mapping errors which then can be fed back to the development team for resolution.


Business as usual flow
Current production input files
Legacy integration platforms (production)
Current production output files
Cloud integration platform (test environment)
Output file from new solution
Process
Automation
Comparison report
edifact - ConslidationReport(24-August-16 AT 05-50-43) [Read-Only] [Comp
FILE
HOME
INSERT
PAGE LAYOUT
FORMULAS
DATA
REVIEW
VIEW
Paste
Clipboard
Arial
10
Font
Wrap Text
Merge & Center
Alignment
General
Number
Conditional Formatting
Format as Table
Styles
C16
1

	A	B	C	D
1	ConsolidatedTPGroup1	TradingPartner	FileName	
2		9429000083164	20160426-021332.EDIFACT.E2E_1	
3		9429000083164	20160426-021332.EDIFACT.E2E_2	
4		9429000083164	20160426-021332.EDIFACT.E2E_3	
5		9429000083164	edifact test.txt_1	
6		9429000083164	edifact test.txt_2	
7		9429000083164	edifact test.txt_3	
8				
9		Structure		
10		Segments	Occurrence 9429000083164 file	Occurrence 9429000083164 file
11		BGM	1	1
12		DTM	3	3
13		FTX	3	3
14		RFF	1	1
15		NAD	3	3
16		LIN	1	1
17		PIA	1	1
18		IMD	1	1
19		QTY	3	3
20		PRI	2	2
21		UNS	1	1
22		MOA	1	1
23			No of repetition of highlighted block = 152	No of repetition of highlighted block = 152

Figure 3 – Bulk data comparison tool for validating output from legacy and new systems

f. Production readiness

Before going live, run various types of tests to ensure smooth cutover to the new system.

System integration testing

- While the development team builds the transformation logic for the initial set of interfaces, establish connection with partners who share the same set of interfaces
- To establish connection, carry out IP whitelisting at both ends. Once IPs have been whitelisted and required configurations have been set up at both ends, run connection testing in the test and production environments with a dummy file
- After completion of connection testing, carry out end-to-end integration testing in the test environment between source and target systems to replicate the message flow taking place in the production environment. Message content and business functionality testing are not part of this test

This marks the completion of system integration testing (SIT).

End-to-end functional testing

- Once SIT is complete, run an end-to-end functional testing (UAT) of all interfaces. However, due to time constraints, UAT can be done on only a few interfaces. Select these 'pilot' interfaces carefully to represent most of the business scenarios
- Identify a set of business users to be involved in functional testing. This phase is also known as user acceptance testing (UAT)
- Test every scenario for the identified pilot interfaces. Testing must ensure that messages are successfully received and processed at target systems
- Once UAT is completed, obtain a sign off on the testing process from the participating business users

Performance testing (concurrency, stress/load testing)

As a best practice, test the new system's performance for "x" times the expected peak load for the next 5 years. In addition to this stress load, test the new platform for user load to ensure the system performs optimally with multi-user logins.

g. Production cutover

Deployment and go-live

- Inform all relevant stakeholders about the cutover details in advance – business cutover date, time required for the cutover, approximate downtime, and rollback plan
- Review and deploy the required code and processes in production at least two days in advance to avoid last minute surprises
- Plan the cutover for a time when the message flow is less. Check that there are no messages in the queue in the old system before switching connections to the nesystem
- Carry out an audit on the old application to check if any messages are still flowing through it

Post go-live support (hypercare)

- The delivery team should handle any issues in the new system after go-live for a defined period called post go-live support or sustain support or warranty support period
- The application management (AM) team should manage the interface after the post go-live support period. The AM team should be actively involved in the post go-live support phase
- Issues faced in the go-live phase should be documented and be part of the handover to the AM team for future reference

Rollback plan

If there is an issue after switching to the new system, all teams should quickly gauge the complexity of the issue and take a conscious call over whether to fix the issue on the fly in production or to roll back to the original connection. If rolled back to the original connection, carry out a sanity check with at least 1 file for each integration. Monitor the rolled back system for the next few hours for any error while the original issue is being fixed.

4. Program governance and management

Program governance and management overarches the group of activities ensuring successful end-to-end implementation of the new platform because it guarantees a support system for the entire program. It provides Project Managers with processes, tools and frameworks to manage a project and identify risks and dependencies.

Program Governance

- Oversight and control
- Establish structure for communication, implementation, and monitoring
- Ensure best practices are followed
- Escalation structure and management
- Performance monitoring and cost control
- CoE (center of excellence) management

Project Management

- Project planning
- Resource planning
- Time tracking
- Scope management
- Rollout management
- Risk mitigation and issue management

Stakeholder Management

- Executive Stakeholder management
- Project Sponsor stakeholder management
- Steering Committee management
- Organizing, monitoring, and nurturing relationships
- Track project metrics

Reporting

- Weekly status reports
- Steerco meeting
- Milestone reporting

Figure 4 – Key aspects of a modernization project

5. Key challenges and solutions

	Challenge	Solution
People	Shortage of skill sets in the market could require significant investment to hire or ramp up the team to learn new platforms or systems	Plan for a learning curve or ramp-up period in the project timelines and set up guidelines for the team to get certified in the new technology
	Lack of understanding of the existing/legacy system	Ensure team members of the legacy system are part of the modernization project to validate existing functionality and assumptions, provide a source of truth, and support the modernization
	Project delays owing to issues in stakeholder onboarding	Ensure transparency in communication and agree on ownership well in advance
	Contingencies and production issues endangering business continuity plan	• Ensure closure signoffs for all phases of the project • Plan knowledge transfer to support team and share technical specifications with them to ensure business as usual after go-live. • Ensure all artifacts are documented for future reference – signoffs, approvals, test cases, UAT results
	Resistance to change	• Engage in a conversation with people who are opposed to the change and understand their concerns • Plan the program in a phased manned to help stakeholders prepare for and support the changes • Communicate the business value of the modernization
Process	Scope creep	Analyze factors like complexity, cost, and benefits before accepting changes being introduced
	Changes in business rules as part of BAU	Align with the BAU team to ensure visibility into progress or planned changes to the business rules
	Lack of adequate system documentation of the legacy system	• Collaborate with application owners and teams to ensure their support when needed • Validate all assumptions with the relevant stakeholders during each phase of the program • Technical specification document is a pre-requisite for cut-over and knowledge transfer to support team
	Scaling regression testing	• Encourage process automation to reduce manual error and effort • Spread testing over multiple days to ensure all business scenarios for each B2B transaction are tested
	Delays in managing production incidents post go-live lead to loss of business and eventually	• Perform liberation test once technical migration is complete
	to a lot of mistrust and escalations	• Audit legacy systems post go-live as a proactive measure to ensure no files are stuck • Ensure proper service portal, email channel or hotline number is enabled and communicated to all stakeholders in the go-live plan and post go-live communications • Set SLAs for incidents based on severity

Technology	Identifying and removing redundant business rules and technical debt	• Ensure an approval workflow is in place to agree on the rules to be removed • Leverage lessons from other modernizations or teams • Add value by consolidating business rules
	Compatibility issues between legacy systems and regression testing tool making it difficult to extract files and automate the regression testing process	• Plan an SOP and communicate to the team in case automated process fails due to system issues • Ensure approval from key stakeholders in case manual testing is agreed upon
	Scalability and performance issues in the new platform	• While considering non-functional requirements, analyze the as-is peak transaction volume and size with some buffer to prepare the system to face these scenarios as part of architecture planning • Plan performance testing before each go-live
	Trading partner infrastructure readiness	• Have the details of trading partner's systems in advance to analyze any compatibility issue • Plan a connection testing phase in parallel to the build phase to avoid delays in go-live
	Security compliance	• Establish guidelines in advance for all security compliances required from the trading partner's system • Follow SOPs set by internal network team in case of exceptions
	Unknowns of the new platform	• Accommodate POCs and accelerators required to analyze functionalities and performance for specific use cases from the existing landscape in the new platform • Work in collaboration with the product team

6. Summary

Middleware modernization projects can be complex and involve considerable challenges. A good program manager must plan ahead, identify the risks and challenges, and ensure smooth transition from legacy systems to the new platform.

In this whitepaper, we have described the phases and stakeholders involved in a project of this nature. The section on key challenges and solutions summarizes key lessons from multiple complex modernization projects. The whitepaper also provides a snapshot of key steps and best practices that can help execute a large-scale modernization with minimum surprises.

About the Authors

Narayana Rao Gedela

Principal, Infosys Consulting

Narayana is a Principal with the CRL practice in Infosys consulting. He has been in the IT industry for over two decades and has strong experience in delivering large complex transformation programs across several verticals for global clients.

Heena Pahuja

Principal, Infosys Consulting

Heena is a Principal with the CRL practice in Infosys consulting. She has 9+ years of IT experience with a strong consulting background in helping customers accelerate their digital automation and transformation journeys.

Infosys®
Navigate your next

For more information, contact askus@infosys.com