The Integration Challenge

Successfully migrating from a legacy engine

Orion Health – White Paper
David Boerner, Solution Engineer

ORION HEALTH

The Integration Challenge

Table of Contents

The Integration Challenge

1. Introduction

The desire healthcare organizations have to adopt cloud and mobile capabilities is increasing, but in order to leverage these technologies, organizations need a contemporary integration platform. However, healthcare organizations are often using older platforms that do not support contemporary integration requirements. In some cases, integration platform vendors are discontinuing support for these platforms or discontinuing the product altogether. Plus, there are currently fewer developers than there were in the past with knowledge of specialized algorithmic programming languages like Monk. This circumstance leaves healthcare organizations with the challenge of how they can move towards advanced technologies on their current older platform.

The integration challenge has evolved dramatically in recent years. As support for outdated legacy engines decreases and the cost to maintain these solutions escalates, the decision to convert to a contemporary product proves to be the most feasible option. This white paper covers an effective five-step approach, and is based on Orion Health's extensive experience of migrating customers from legacy engines.

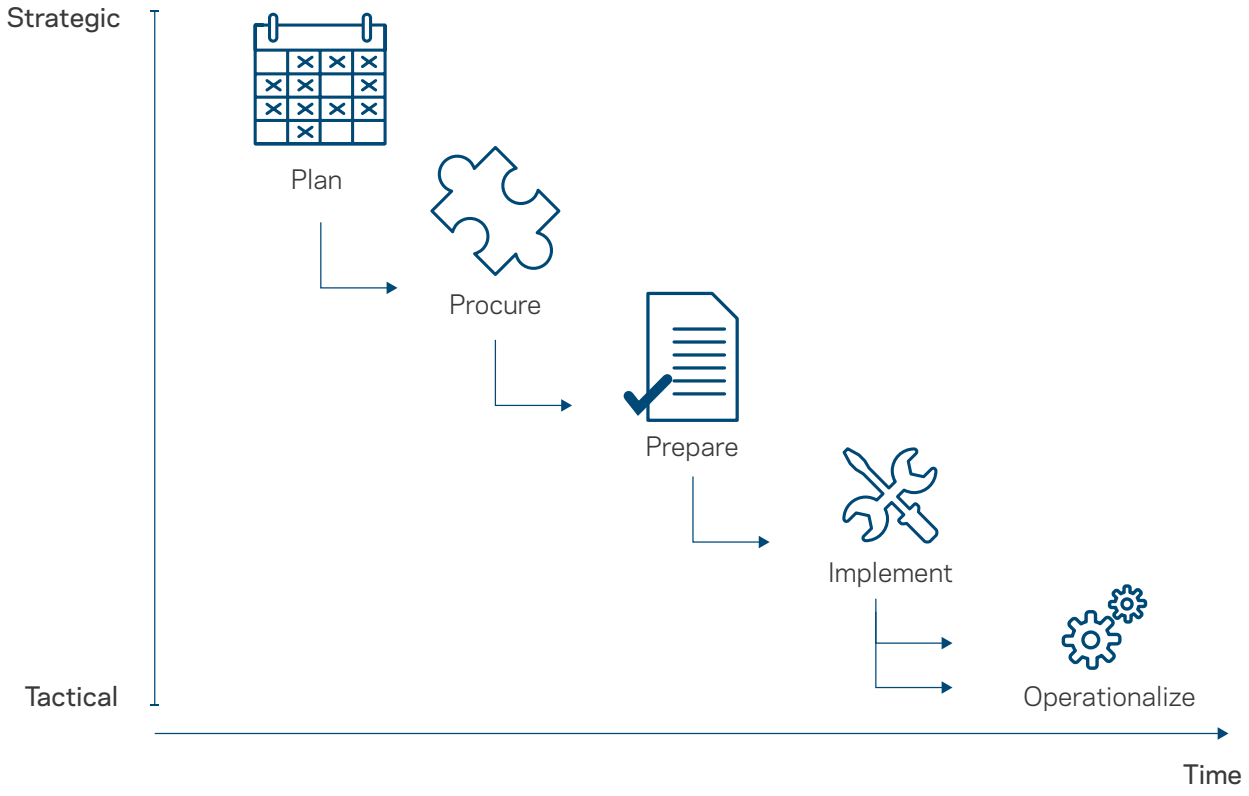


Figure 1: Five Steps Migration Process

2. Migration Steps

Step 1: Plan

The first migration step should be to formulate an individual plan specific to the healthcare organization. The relevant components of the project plan should include the scope, technical requirements, dependencies, timeframe and a planning checklist.

Scope

The scope of a migration project may be quite narrow, and limited only to replacing the products that are being sunsetted and re-implementing the exact interfaces that are impacted.

We find many organizations choose to take advantage of a fresh start on a new product to improve existing interfaces, deploy new interfaces and/or consolidate interfaces onto a single platform.

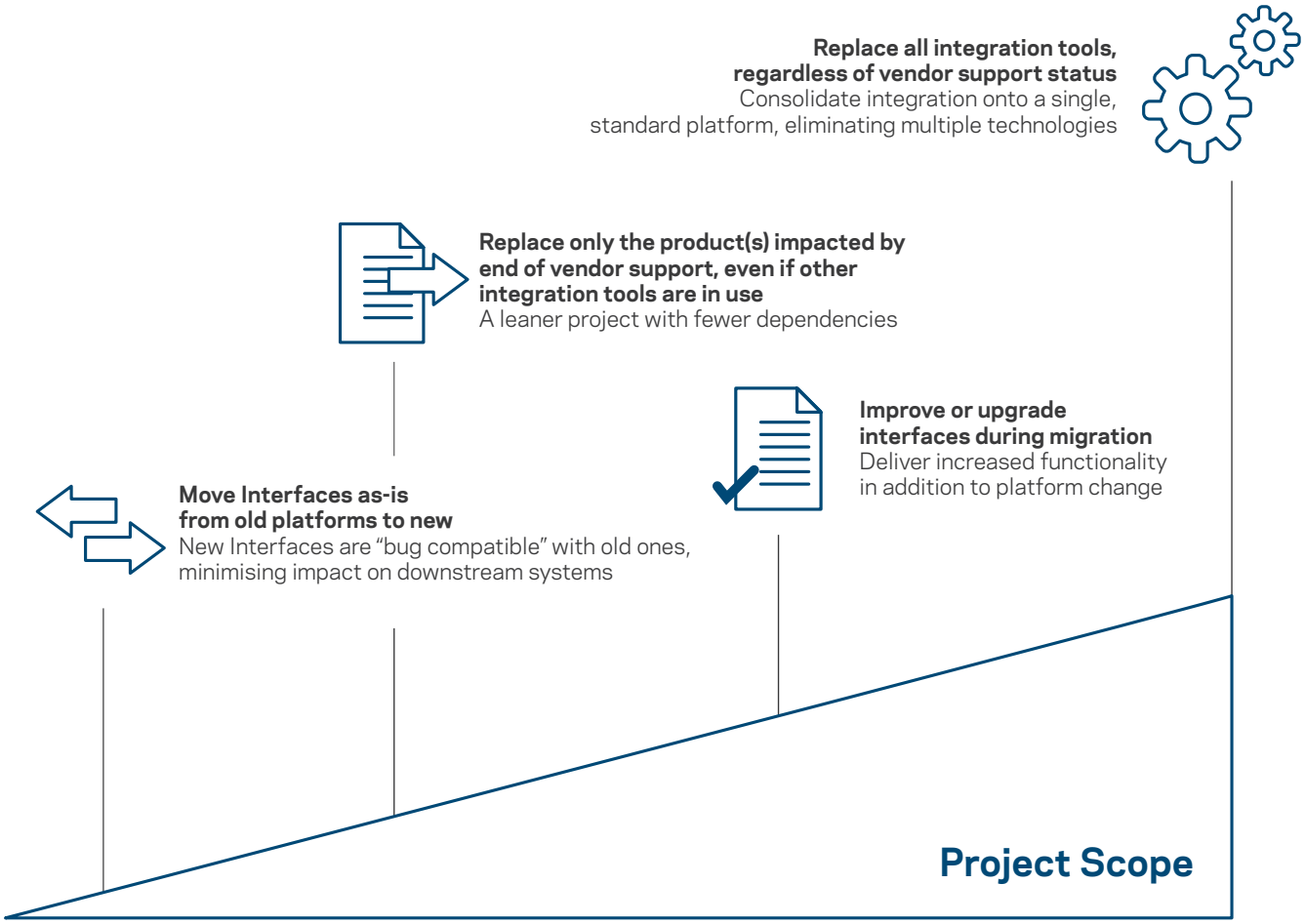


Figure 2: Project Scope

As scoping decisions are finalized, a clear picture will emerge of how many interfaces need to be implemented on the new platform. This number will become the most basic measure of the project scope and help an organization determine the size and expertise required.

The Integration Challenge

Technical Requirements

Defining the functional and non-functional requirements at the beginning of the project will guide the project team's decisions and set the stage for a successful outcome. High-level requirements should answer the following questions:

- Which clinical systems need to connect to which other systems?
- What technical environments will the integration tool run in (i.e., hardware, software, operating system)?
- How many environments are needed to support the organization's development process (e.g., development, test, production)?
- What existing policies around security, auditing, disaster recovery and uptime will the new technology need to conform to?
- What are the performance metrics in terms of scalability, throughput and uptime?
- What future considerations need to be taken into account?
- What other technologies could be folded into the new engine? Examples include separate FTP movers, file listeners, email generators, web service utilities, etc.
- What messaging standards do the interfaces adhere to?
- What messaging, protocol and security standards need to be supported? Listed in the chart are examples of built-in protocols:

Protocol	Secure Option
TCP/IP	Yes
HTTP	Yes
Web Services - Call	Yes
Web Services - Host	Yes
Database	Authenticated
Email	No
JMS, MSMQ, IBMMQ	No
Flat or Zip File	No
FTP Client	SFTP and FTPS

Dependencies

As a project is not executed in a vacuum, it is important to take dependencies into account. Dependencies in a project may result from other projects. A growing trend in the industry involves combining engine migration with other system migration initiativies. We advise clients to roll their legacy migration out first or concurrently to eliminate interface re-work.

Planning Checklist

Planning is a critical phase in any project. This checklist summarizes the necessary steps to planning your migration.

- ☐ Determine project scope: broad, narrow or in between using the number of interfaces to be completed as your guideline
- ☐ Document key technical requirements
- ☐ Determine project dependencies
- ☐ Plan the project completion dates
- ☐ Establish your resource requirements using the number of interfaces and the project timeline as your guideline.

Step 2: Procure

Once a project plan is developed, the next step is to identify a replacement engine. In this step, the engine should be evaluated based on product capabilities, service offerings and vendor experience. Another important factor is the Total Cost of Ownership (TCO) of the new platform compared to your existing platform. Some vendors will charge ongoing support for features that are not required by the healthcare organization. Additionally, organizations should take into consideration services such as the type of initial and ongoing training provided and ease of obtaining support.

When looking for a new integration engine, vendors need to consider the ongoing client support. What is included in the support and the maintenance agreement? Does the new integration engine vendor provide initial and ongoing training of the product? Is there a certification program available? How is the customer support accessed? Through email, phone or online chat. Is there an online ticketing system? Which enables easy tracing of issues. Ongoing client support is an important factor to consider when comparing integration engine vendors.

Evaluation Process

The evaluation criteria should come directly from the requirements developed during the planning phase. In order to see whether a particular solution will meet your requirements, it is common to evaluate vendors by asking for a "Proof of Concept" to demonstrate their ability to migrate existing interfaces.

Procurement Checklist

- ☐ Develop evaluation criteria based on project requirements
- ☐ Prioritize requirements into "Must Have" and "Nice to Have" categories so that a "best fit" solution can be identified
- ☐ Identity interface engine
- ☐ Evaluate integration engine according to consistent criteria
- ☐ Request and evaluate "Proof of Concept" demonstrations

Partnering with vendors that have successfully migrated legacy engines and have performed the appropriate gap analysis on components and functionality will accelerate your migration.

Another issue to consider is the requirements for developers – i.e., do they need specialized algorithmic programming languages? Some older languages such as Monk have fewer developers available with this knowledge. Whereas, JavaScript is a modern language and there are multiple developers available.

A sample analysis of components is provided below:

eGate/SRE	JCAPS	Rhapsody
ETD	OTD	Message Definition (.s3d)
Collaboration	Connectivity Map	Route
Eway	External sys (adapter)	Communication Point (Comm. Pt.)
Collab Rule	Java Collab def (JCD)	Mapper definition file (.mdf)
Enterprise Designer	Netbeans	Rhapsody IDE
Enterprise Monitor	Enterprise Manager	Web Management Console (WMC)
ETD Tester	OTD Tester	EDI Analyzer/ Explorer
<programmatic>	<programmatic>	Conditional Connector
Registry Host	Repository	Rhapsody Engine (Config)
Monk Function or Java Method	Java Method	Filters
Shema/ Component export file	Project export (.zip)	.rlc file

Step 3: Prepare

Provisioning

The most important preparation migration task is to ensure that proper hardware, software and licenses are available.

It may be possible to repurpose servers from the legacy engine for the new engine; however, most organizations prefer to run the engine on newly acquired hardware for the following reasons:

- Typically, there is a period of time in which the legacy and new integration engine must run concurrently, creating the need for separate environments.
- If the engine is running on older hardware, implementing a new interfacing solution is a good opportunity to update the hardware at a relatively low incremental cost.
- Another alternative is to move the solution to a virtual environment, therefore utilizing existing hardware, yet providing a portable mechanism for easier migration.

Staffing

The scope and timeline form the basis for determining the appropriate experience and number of staff for the entire project. A typical interface project team can include:

- Project manager
- Interface analysts/developers
- Application specialists (often part-time subject matter experts)
- Quality Assurance analysts
- Operations analysts

Training

Training for analysts, developers, and operational staff also takes place during the preparation stage. Ensure the vendor offers a combination of product documentation, online self-paced instruction, onsite training and continued mentoring.

Prioritization

Each interface should be prioritized and (optionally) classified according to the difficulty of the project. Often, clients can realize a substantial consolidation when migrating, due to differing integration paradigms. It is also advisable to highlight interfaces to be added due to increased functionality provided by the new integration platform.

"Rhapsody's ease of use combined with powerful functionality allowed us to migrate quickly while simultaneously enhancing integration between PCH applications."

Kevin Allen, Senior Integration Analyst, Phoenix Children's Hospital (PCH)

The project team should develop a strategy for scheduling interfaces for migration. They may prefer to schedule high-priority interfaces first and lower priority interfaces later. Or, they may start with easier interfaces and work up to harder ones as they gain familiarity with the new tools.

- ☐ Provision hardware, software, and licenses required for the project
- ☐ Assemble an appropriately sized project team
- ☐ Conduct training for all team members
- ☐ List and prioritize all interfaces in scope for migration

Preparation Checklist

Step 4: Implement

Migration

Working through the previously created prioritized list of interfaces, the project team can migrate each interface individually or in bulk. Typically, this consists of the following steps, documented in the illustration below:

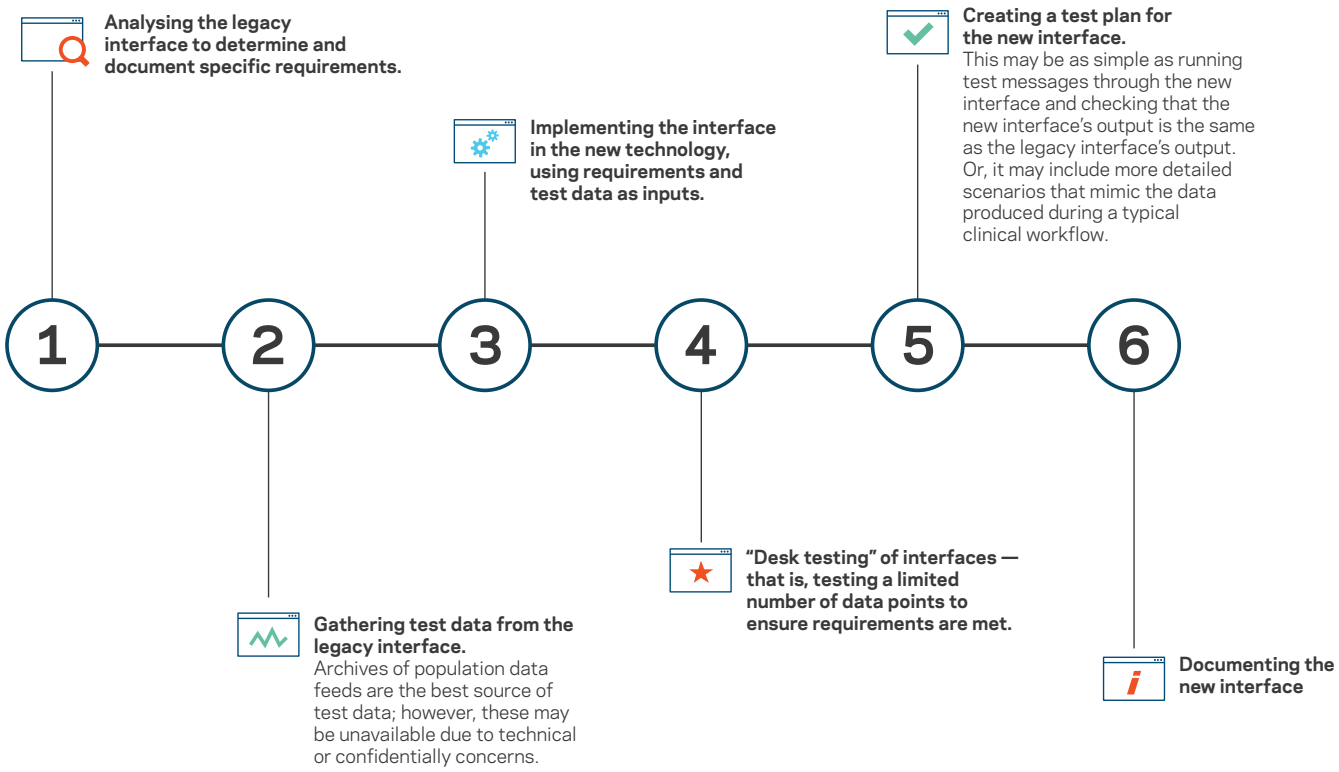


Figure 4: Integration Migration Steps

The Integration Challenge

Other Considerations

In addition to migrating individual interfaces, there will be non-interface items that need to be implemented. These include such items as:

- Development of backup and recovery plans in a highly available clustered, disaster-proofed environment
- Monitoring strategies related to proactive alerting
- Development of processes to deal with system event notifications
- Maintenance of users, roles and security settings
- Paradigm shift related to modernized engine architecture

"With eGate some of our biggest challenges were monitoring the interfaces and operational reporting. These pain points have been eliminated with Rhapsody's configurable web management console, which enables us to proactively manage our interfaces and report granular information on each."

Corey Smith, Manager IT Integration, University Hospitals Management Service Center

---

Testing

The first step in testing an interface is functional testing, to prove that the interface meets the requirements defined at the beginning of the project and to ensure correctness of outputs. Look for an integration engine that streamlines this process by enabling granular component testing, which can eliminate time-intensive end-to-end test cycles. The next step is to send a high volume of messages through the interface to ensure adequate performance and correct outputs under expected production load volumes.

"Our organization has been quite satisfied with the dependability and speed of development turnaround time with the Orion Rhapsody Integration Engine."

Harvey Wittmayer, IT Application Manager, Trinity Health

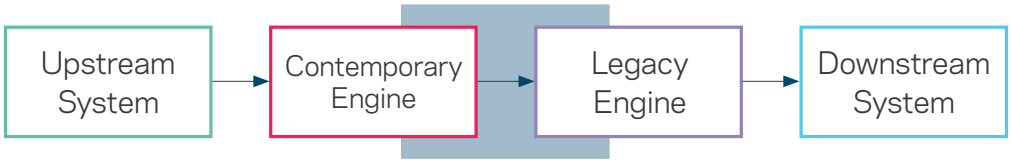
The Integration Challenge

Orion Health has developed a unique approach which can help to automate this process by creating real-time reports on differences between messages, and downstream systems receiving messages from a contemporary engine compared to the legacy engines.

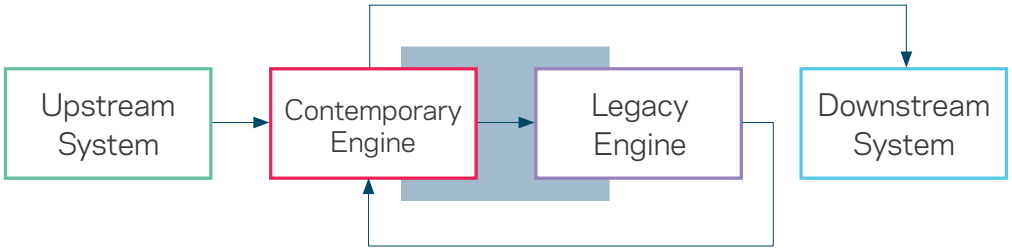
This allows for seamless transition between the two systems, as well as visibility into the effectiveness of the new translations.

An explanation of the diagram follows:

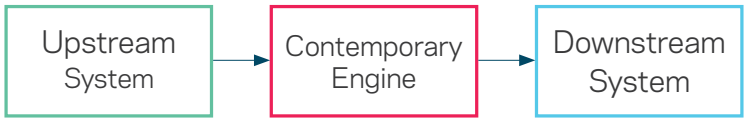
1. Contemporary Engine pass through



2. Live with loopback



3. Retire legacy engine



1. Contemporary engine is inserted upstream of the legacy engine as a pass-through to establish quick wins of connectivity.

2. Legacy mappings are replicated in the contemporary engine; they are run through compare filters to validate that they match legacy mappings. Legacy downstream connections now flow through the contemporary engine, establishing connectivity downstream.

3. After UAT is complete, downstream connections are switched directly to the contemporary engine feeds and the legacy system is retired.

Implementation Checklist

- ☐ Gather requirements then build, test and document each interface
- ☐ Conduct testing and defect remediation for all interfaces

Step 5: Operationalize

Promote to Production

To ensure best practices, promotion should be done according to a defined, repeatable process. The contemporary engine's migration solution allows for the promotion of individual or groups of interfaces, depending on the client's preference.

After the migration is complete, the interface should be monitored closely for a "burn-in" period to ensure that unexpected failures are detected promptly. The contemporary engine's Message Compare filter is ideal to automate this period, to ensure downstream systems experience a seamless transition.

Legacy Engine Retirement

When the operational team feels certain that the new interface is performing correctly, they may retire the engine by shutting down each legacy connection. After all of the legacy connections have been shut down, it is safe to de-commission the engine.

Operational Checklist

- ☐ Promote new interfaces to production according to a defined, documented process
- ☐ Retire interfaces by turning off connections
- ☐ When all interfaces have been shutdown, retire engine

3. Conclusion

The integration challenge of replacing a legacy interface engine may seem like a monumental task. Especially as support for outdated legacy engines decreases, the decision to convert to a modern product proves to be the most feasible option. By carefully implementing the five phases outlined in this paper, an organization will be future-proofing with the investment of a contemporary platform. A contemporary product provides a modern integration engine – which can provide healthcare organizations with aspirations to adopt cloud and mobile capabilities – the tools to implement these.

4. Learn More

To learn more about how Orion Health can help your organization replace a legacy integration platform with Orion Health™ Rhapsody® Integration Engine, please visit **www.orionhealth.com/rhapsody-integration-engine**