Interface Engine Migration Guide: From Oracle e*Gate to InterSystems Ensemble

Table of Content

Introduction

With DataGate/eGate at the end-of-life mark, you may be one of many hospitals and health systems trying to replace eGate engines with a newer technology. Once you've selected the interface engine you plan to move to, you know there is a lot of work ahead!

The biggest investment isn't the software licensing or your incremental hardware costs. Your most costly line items will be the development and implementation time – labor costs.

The goal of this paper is to help in the design of a migration project from eGate to InterSystems Ensemble Integration Engine with consideration to the various elements you'll need to include. We'll recommend a workflow to improve team productivity and reduce those labor costs. In addition, we'll offer guidance on how to accomplish this using **Caristix Workgroup** software and the **eGate and Ensemble connectors**.

The Trial and Error Loop

It's pretty common in the industry to have developers jump right into the code. Requirements are often incomplete or partial and the data to be exchanged is not well understood. With this incomplete process, developers start to code and send messages to the new interface. Issues start to pop up rapidly. The code is then edited for a quick fix. Though issues may be fixed quickly, others appear and you're off and into the *trial and error loop* in a reiterative rinse and repeat cycle.

The problem with this approach is that the number of issues and the time required to make corrections is unknown. Your team keeps working until no new issues are discovered, in other words, until the end of the project. In addition, if your test coverage is not exhaustive, chances are issues will not be discovered until the interface has moved into production.

Workflow Recommendations

The following proposed workflow describes a technique to speed up eGate to Ensemble migration project execution and increase the quality of deliverables.

There are five stages in interface migration:
1 - Gather Interface Requirements
2 - Setup the New Interface
3 - Migrate the Interface
4 - Validate the New Interface
5 - Move into Production

This suggested workflow can scale to several hundred up to potentially thousands of interfaces. We'll focus on how several manual tasks can be automated to speed up the process while reducing error and producing higher quality interfaces and documentation. We'll also cover how to achieve the level of automation required to effectively validate new interfaces while increasing the productivity of your migration team using InterSystems Ensemble and **Caristix Workgroup.**

These recommendations remain valid even if you are not using the tools described or are using different integration engine technologies. In general, this process can still help you structure and standardize migration projects.

For simplicity, the recommendations do not consider the usage of canonical or standardized models. A few steps would vary but the overall process is compatible. Please do not hesitate to contact us for more details. In addition, the recommendations do not address separate development and production environments, nor discuss network issues.

Step 1: Gather Interface Requirements

To write new interface code and validate new interfaces, the first step is to clearly define the interface requirements. Identify and answer any questions related to what the current interface running on the legacy infrastructure accomplishes. Often, interfaces are not well documented and several business rules and data logic are not necessarily known. Just looking at the interface code will not be enough to gather requirements.

You need to understand the system workflows, the data to be exchanged, the format of that data, and the data semantics.

1. **What data is exchanged?**

   Once the overall workflows are defined, you'll know exactly what data is sent and what data is expected on the receiving end. Pay attention to the fact that sometimes the data semantics differ on both sides of the interface.

2. **What data formats are used?**

   Now that you know what data will be transmitted, you need to understand the sending system data format and the receiving system expected data format. Data formats would usually be different on each side of the interface. Understanding inbound and outbound message formats helps you list transformation requirements. Getting a detailed conformance profile (HL7 specification) for each system involved in the interface is a good start.

   Here are two downloadable examples of an interface spec or profile, one in Word and the second in Excel.

   eGate also uses ETD (Event Type Definitions) describing message formats, which serves as a good source as it contains customizations. Export them to a document to work with and build documentation.

   Note: Keep in mind that building HL7 specifications out of a single or just a few message examples is not enough. You need to understand the various message customizations and semantic variances applied to specific clinical and sending application flows.

3. **What data pieces must be transformed and how?**

   HL7 is a hairy beast. To tame it, more than pipe counting is needed. The HL7 standard is so flexible that each system has its own interpretation of the standard. It's not just the Z-segments, customized fields and tables are also very common. To get a complete list of transformations required, a thorough gap analysis is key.

What is an HL7 Specification?

As HL7 International® states in the v2.5 standard definition:

"An HL7 message profile (or HL7 specification) is an unambiguous specification of one or more standard HL7 messages that have been analyzed for a particular use case. It prescribes a set of precise constraints upon one or more standard HL7 messages."

In other words, a specification is a description of the data and messages that an interface sends and/or receives. The description covers:

- Trigger events supported
- Data format (segments, fields and components descriptions)
- Data semantics
- Message acknowledgment responsibilities

The description must be clear and precise enough so that it acts as a set of requirements.

Getting the Specification from Vendors is *Not* Enough

Most of the time, the first reflex is to contact the system vendor to get the HL7 spec of their system. Unfortunately, the quality of vendor specification varies. While some are quite detailed and reliable, others are incomplete or just don't exist.

Also, those specifications do not always represent the specification of the actual implementation of the system in **your** organization. The vendor spec is generic. The customizations created to match your organization's reality will not be available in that generic spec.

The vendor spec, when available, is a good start but you need to make sure it is complete and identify any customization that has been applied.

Gap Analysis

Once you have HL7 specifications (conformance profiles) for your source and destination systems, capture a list of all the differences existing between the two systems to be interfaced in a requirements document. You create this list by conducting a gap analysis, which will tell interface developers what's missing and what needs to be bridged by the interface. In essence, a gap analysis captures the differences in messaging between the sending and the receiving system so the systems can exchange data and understand it.

Many analysts develop their own gap analysis templates in Microsoft Word or Excel. To fill in templates, they look at messages, run queries when they can, and manually document their findings. This can be a fairly onerous process. You can download our free Caristix Gap Analysis Template to help you with this process.

- If this is done using the vendor spec, make sure it contains everything needed and it describes the actual implementation of the system.

Consider porting the vendor spec to a generic format so it can be easily compared.

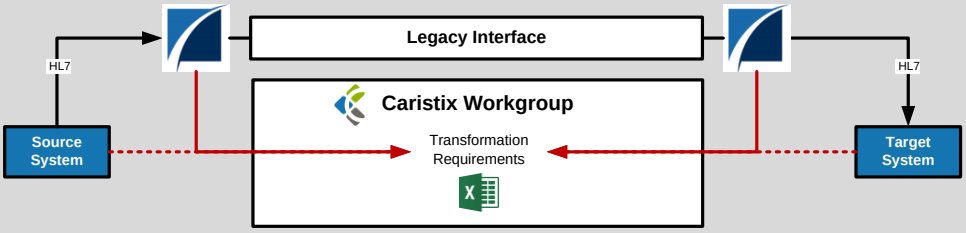
- For the best results, base the analysis on real-world messages. However, the analysis is complete only if the messages analyzed cover all workflows.

Without a gap analysis that details your requirements, you'll end up implementing a generic interface with only some of the transformation, mapping and business needs. You'll also end up wasting time, money, and effort troubleshooting after going live. With a gap analysis, you can avoid extended go-live periods, significant maintenance at increased cost, and unhappy clinician end-users who are unable to access the data they need to deliver appropriate patient care.

Detect gaps early in the project and save on costly iterations. Time and effort spent on a thorough gap analysis will pay off later during the project.

Automating the Process

Caristix Workgroup software helps you automate the requirements gathering process. It increases the accuracy of findings and reduces the time required to manually analyze the various artifacts.



With **Caristix Workgroup**, connect to the eGate engine and extract schemas and messages, and create a full HL7 specification. The entire process takes just a few minutes to complete.

The generated specifications describe the real thing: the actual systems sending and receiving HL7 data, including any segment, field or code set customizations.

The results can also be used to create several project deliverables:

- **Message definitions for integration engines**: An HL7 specification can be automatically converted to message schemas that Ensemble can import (read more about this in the *Build the Interface* section).

- **Interface documentation**: Export the HL7 specification to a Word document. The documentation can now be made available to anyone who requires it. In addition, the documentation is now in sync with the Ensemble message definition.

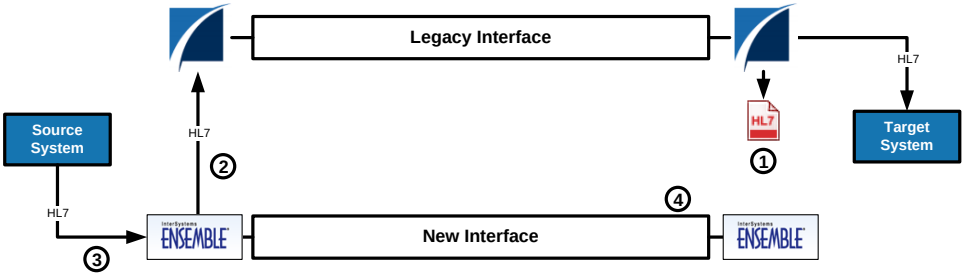
Once you have the HL7 specifications, use **Caristix Workgroup** to run comparisons and find the gaps (what transformations must be implemented). **Caristix Workgroup** detects differences between a large set of inbound and outbound messages. Field length, optionality, repeatability, field content and data mapping differences are detected automatically. Thousands of messages can be analyzed in one pass for all covered workflows in just a few seconds.

Caristix Workgroup generates a complete gap analysis report which, with other business requirement you might have, can be used for interface requirements.

Step 2: Setup the New Interface

In parallel to the interface requirements gathering process, the new interface on Ensemble can be configured so that it manages the message feed.

The goal is to let Ensemble receive messages and forward them to the eGate interface. Having the new technology in the front end of the communication flow will ease the migration and the new interface testing.



1. If not already configured, make sure any message sent is recorded in a file or in a database. It will be used later during the validation phase to compare the legacy and new interface outputs.

2. Configure the new interface to redirect any traffic coming from the sending system to the legacy interface. Don't modify the legacy interface. It should continue to acknowledge, transmit and transform all messages as it has been doing.

3. Redirect the source system to send HL7 messages to Ensemble (and forwarded to eGate).

4. Create the new interface so the traffic coming from the sending system is also sent to this new interface.

Step 3: Migrate the Interface

Everything is ready to start the interface code migration. If you have collected all the business and coding requirements previously mentioned, migration should be straightforward.

Build the Message Definitions

Message schemas must be defined. This requires an inbound and an outbound message shemas. Transformations will be implemented to create a bridge between both schemas.

To create new message schemas, use the artifacts created during the requirements gathering step. You can also re-use the specification from the eGate Event Trigger Definition (ETD) module. Use it to create the Ensemble schema definitions. It's complete and it represents the real state.

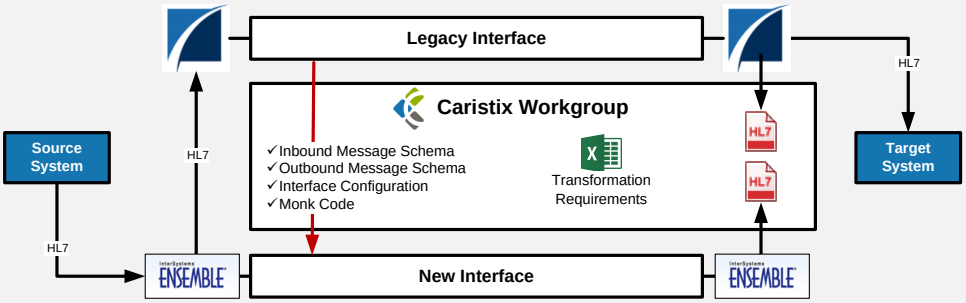
Migrate Monk Code

This next step may be the trickiest part you'll encounter. Complex interfaces can contain a fair amount of code and sometimes suffer from several years of enhancements, tweaking and, possibly patching. Valid requirements implemented at the time might not be applicable today. Also, if you're planning to re-engineer the way data is exchanged and/or transformations are done (message standardization, canonical model or other), some code will no longer serve that purpose. Even more, each programming language has its own strengths and ways to accomplish similar tasks. Consider leveraging the strength of the newly acquired technology instead of carrying over non-functional artifacts and outdated work from previous interfaces.

The Monk code is important for one particular reason: to validate and complete the requirements list. Review it to make sure the new interface will include all the details added along the history of the legacy interface's use to improve the quality of the data exchange in the eco-system.

Automating the Process

Caristix Workgroup offers a way to automate error prone manual tasks developers and analysts must do during interface migrations.



With **Caristix Workgroup**, you can convert the eGate message definitions to the Ensemble message schemas. Create the message schema in Ensemble and start using it within a minute or so. Get it right the first time! If customization is required, customize through the intuitive UI provided by Caristix and automatically generate all the documentation needed.

Caristix can convert the message definition retrieved from eGate during requirement gathering to the Ensemble message definition format. Caristix can export the eGate schemas as XML files that Ensemble will import as message schema.

Step 4: Validate the Interface

Validation is the most important and critical step in the process. Unfortunately, it's the step that many teams bypass. Be sure to adequately plan this critical part of the process to guarantee functional delivery through thorough testing: "Find early, fix cheaply". Test before promoting the interface outside of the development environment. Time not spent on testing and validation will be spent later troubleshooting and fixing errors on a live interface.

Manual Testing

Each business and data transformation requirement should have one or more associated test cases so that you can validate that all requirements are covered by the new interface.

To test the new interface:

1. **Build test cases:** Make sure tests cover all requirements and workflows. Build a set of messages so you are confident that requirements are covered. This aids test coverage, which is important in measuring risk. Make sure those messages (test cases) are saved so that they can be re-run whenever needed.

2. **Test the legacy interface:** Run test cases on the old legacy interface and make sure the results are saved.

3. **Test the new interface:** Run the same exact tests on the new interface and save the results in the same format as the legacy interface results.

4. **Validate results:** Compare results of the same test cases on both interfaces. The expected results should be identical. Manual validation can be challenging depending on the number of test cases and testing frequency. Consider automating this step.

5. **Fix** the interface if needed

6. **Go back to step 1:** Expect requirements and test cases to evolve with time. A good interface testing infrastructure makes it easy to handle changes. It also makes it easy to rerun and validate all test cases as frequently as needed.

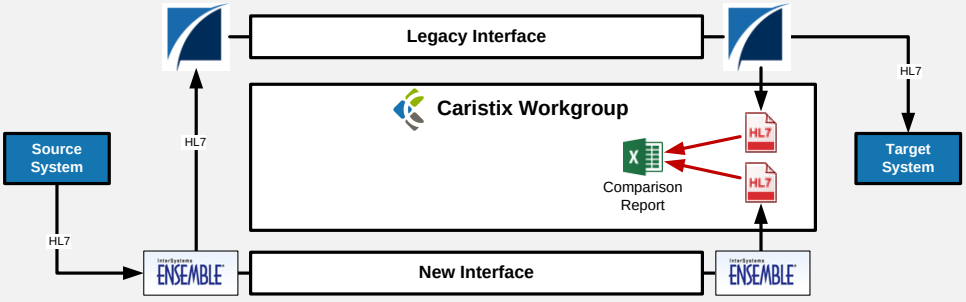
Automating the Process

Thoroughly testing a newly created interface includes a lot of repetitive work. Manual validation is error prone and effort intensive. If you change the code, you have to redo it all again. Errors you don't catch will make their way into production.

Caristix Workgroup offers a wide set of functionalities to validate an HL7 interface. For each test, reports can be generated to ease team collaboration and enforce the rigor of the quality process in place.

Message Comparison

Caristix Workgroup lets you compare the output of the new interface with the output of the legacy interface for the same set of HL7 transactions. Validate that the output is the same for both. The message comparison is HL7 savvy and compares actual HL7 structures (not strings). It allows for hiding differences in fields and/or segments for which you don't need notification. Find messages to compare even if messages in files are not in the same order. **Caristix Workgroup** makes it quick and easy to compare large sets of messages and highlight any differences.



Conformance Validation

Validate that the output messages conform to the HL7 spec. The conformance validation spots any conformance issue and notifies you. Again, use a large set of messages so you are confident all cases are covered.

Field Validation Rules

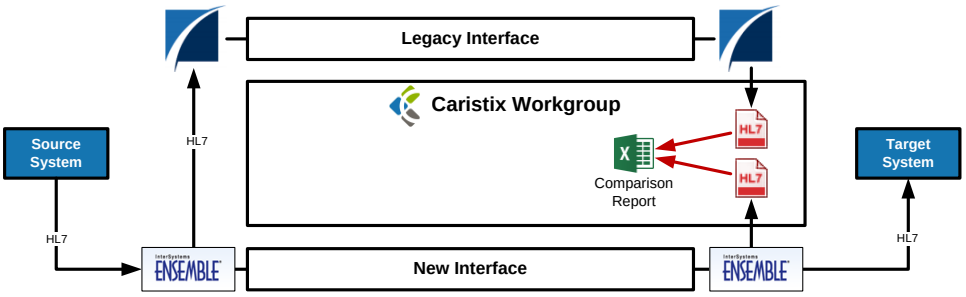
Validate specific business rules, data format and data logic on fields using the field validation rules. Workgroup offers an advanced validation language to help build sophisticated rules to validate complex business requirements.

Test Automation

Improve your team's productivity by automating all test cases. This allows for system simulation sending and/or receiving messages and applies validation rules to the exchanged data. Messages or pieces of messages can be generated at runtime, reducing the effort required to maintain test cases and improving test coverage. When you make any changes, re-run all units, regression and functional tests pressing the RUN button. The Execution report is automatically generated and you can validate all results.

Step 5: Move into Production

Once the interface is validated and you're ready to retire the legacy interface, stop the delivery of messages from the legacy interface and let the new interface take over the functionality of delivery to the target system.



Making Your Team Productive

As effort spent per interface varies depending on the organization and the interface complexity, labor costs will vary as well. As guidelines, here is some data users shared with us.

Table 1: Effort and cost per interface

	Task	Effort (hr)	Cost ($)
Without Caristix	Gather Requirements and Build Message Definitions	20	$2000
	Validate the Interface	32	$3200
	Total (per interface):	**52**	**$5200**
With Caristix	Gather Requirements and Build Message Definitions	2	$200
	Validate the Interface	10	$1000
	Total (per interface):	**12**	**1200**
	Savings (per interface)*:	**40**	**$4000**

**Based on a few phases of the recommended workflow and not accounting for documentation generation and interface maintenance. Estimated FTE cost of $100/hour.*

Feedback

We'd love to hear from you. Did the information in this paper help you with your project? Have we left anything out? Is there more or other information that would have been valuable?

Let us know how your project went. We'd like to share that information with others trying to migrate their data exchange infrastructure.

Resources and Contributors

Authors

Jean-Luc Morin
Margo Ellis

Invaluable contributions from two members of the Caristix software development team, Dominic Bérubé and Maxime Dupont.

About Caristix

Caristix has developed a software suite to automate manual interface work. Our software reads HL7 data and outputs a list of interface requirements. As a result, Caristix software can reduce months of work to a few days.

Learn more at http://caristix.com.

Follow Us

twitter.com/caristix
linkedin.com/company/caristix
caristix.com/blog

Contact Us

Caristix
1400 St-Jean-Baptiste Avenue, Suite 204
Quebec City, QC, Canada G2E 5B7
Tel: 1-877-872-0027

info@caristix.com
http://caristix.com