hopp

migrating complex data.
faster, cheaper, better.

Why migFx and not ETL

for complex data migration

What is the problem

The challenge in any complex data migration is not just to move data from an old system into a new system.

The real challenge is to do it a way that ensures the business logic is moved or transformed in a way, consistent with the new system.

Understanding that it is relationships and dependencies in data that matters, is the topic of this paper.

We make the case for using a solution already specifically developed for complex data migrations to save time, money and ensure quality of outcome.

What is the question

What are the first questions you consider when faced with a complex data migration?

- Do I have the right people?
- Do I have the data needed?
- Do I have the right methodology?
- .. and do I have the right tools?

While all these questions are important and, in some way linked, we believe that making the right choice on each will benefit the others by providing knowledge and structure to the project.

A complex data migration could be anything with not just size but most important complexity. Complexity is dependencies, relationships and business rules - for instance when migrating from a large enterprise legacy system to a new ERP style system.

In this paper we will provide our thoughts on the importance of choosing a proven tool allowing you to move business logic in an agile and iterative process and not only tables in defined flows.

Summary

There may be reasons why someone would choose to spend time and effort to build a bespoke data migration solution but. in general good industry practice would be to buy specialised tools for this and not attempt to build what already exists.

We know from experience across the industry that building a flexible, comprehensive and robust data migration solution requires significant investment in time and resources and that key design choices must be made to achieve the agile and iterative functionality needed - some of which are not possible using flow based ETL tools like Microsoft Sql Server Integration Services (SSIS) or others.

Many underestimate the task and hence end up with suboptimal solutions or direct failures.

hopp tech ltd
86-90 Paul Street
London EC2A 4NE
United Kingdom

t +44 (0)203 322 1718
e info@hopp.tech
w www.hopp.tech

Registered at the above address
Company number: 10421313

In summary, we believe that everyone attempting to build a data migration solution based on ETL, will

- Build first gen solution vs buy mature solution
- Spend more time and resources than needed
- Achieve an inferior solution
- End up with a bespoke setup and key personnel dependency
- Not have a proven and transparent onboarding offer for new customers

We strongly recommend you consider applying a dedicated data migration tool like our migFx instead of starting internal development and to get a clearer understanding of the pros and cons of the different solutions.

migFx - key features

An End-to-end solution for complex Data Migrations

Our solution boasts all the tools you'll need to successfully complete a complex data migration.

Powerful, easy-to-use graphical user interface

A Windows application that allows you to map data, define business rules, create data validation, perform tests and track events.

Logical Business Objects

The concept of Business Objects means that you can stay focused on pure business logic throughout.

Consistent application of dependencies, validation and rules

Quality is built-in. By consistent application of mapping, rules and validation into the actual code, errors in the process are minimised.

Before you decide which path to take, you should have a look at existing solutions like migFx – and reflect on the task of building something similar or better.

We are confident that a project based on a standard solution will be up and running in record time, delivering usable output and useful insight before you even get started on your own design.

Here are the why's!

Just as you must build a data migration solution founded on an ETL tools like SSIS or otherwise, migFx is an already built and battle tested data migration solution founded on standard Microsoft infrastructure (Sql Server, .NET, c# and Visual Studio).

To build something yourself you will need ETL skilled resources with deep understanding of data migration issues to build and maintain an ETL based solution. However, as migFx is already developed and is continuously maintained and evolved by hopp, the equivalent of these ETL skills are not required. The core skill in both cases is knowledge of the business, data and systems involved.

migFx requires only these skills, no other technical skills are required. With an ETL based solution, weight is on technical ETL skills whereas the core skill of system and data knowledge is secondary. This is not an ideal approach.

Building a robust data migration solution is not a trivial undertaking. Whatever the foundation (ETL or otherwise), it is a challenge to build a migration solution that remains stable, recognizable, robust and maintainable over years of repeated data migrations into a continuously evolving target system.

migFx is such a solution and has matured for over 10 years in critical, high-complexity data migrations very similar in nature to what you will encounter: Repeated migrations into (and out of) the same target system, competing with other providers offering similar services. The capability to safely migrate data with a robust solution into your system is an important factor when negotiating with potential new customers.

In other words: Don't underestimate the task of developing a robust and lasting data migration solution and don't build it when you can buy it.

Core qualities of migFx

Below are some of the core differences between migFx and any solution build on an ETL tool like SSIS. While there are many more differences, the focus is on valuable aspects of migFx that will be challenging and, in many instances, probably so contrary to the fundamental function of an ETL tool that they in practical reality are impossible to implement in an ETL based solution.

Unit-of-migration: Business Objects

Any solution built on an ETL tool, will inherently be table-centric, flowing records from tables (or other data-sources) across from source to target, enriching/validating on the way with data from other tables/data-sources. Much like a Ford assembly line approach.

What makes migFx different and unique is that migFx does not migrate tables and records in this way. Instead, migFx is 100% built on the concept of Business Objects that are migrated as complete and autonomous units. When migFx executes, one Business Object at a time is extracted from the source in its entirety with data from all concerning tables and migrated as one, complete unit. To stay in the automobile metaphor: In contrast to the assembly-line approach, migFx completely builds one car at a time.

The Business Objects are the foundation of the design of migFx. They bring a long list of benefits to the table, here are some (but far from all):

- migFx can migrate Business Objects in granularity down to a single Business Object. This is invaluable both in the project lifetime, and especially in a critical go-live situation, where migFx can safely and quickly iterate a sub-set of Business Objects to clear up a problem as opposed to having to rerun the entire migration in a flow based solution, where it is impossible to isolate single Business Objects for iteration
- migFx will recursively resolve dependencies between Business Objects and automatically ensure correct order of migration. Again, this is valuable in the lifetime of the project and especially in go-live where the iteration of a sub-set automatically will recursively iterate all dependent Business Objects – and only these
- migFx correlates all data and messages (events) from the migration to Business Objects. This enables the migFx Tracker application to present a comprehensive view of every single Business Object: All Events raised, data extracted from the source and data sent on to the target. This provides unique and invaluable support in the lifetime of a migration project. From the iterative development, over the support for user acceptance test to complete documentation that can be archived after go-live
- The Business Objects provide a valuable abstraction from the complicated world of tables and fields in the target system. Throughout the lifetime of any migration project, everybody – from the experts in the target system, the people identifying the requirements, to the migration team and the business users involved in test – will naturally have a common understanding and terminology based on the Business Objects

The benefits of the Business Object approach cannot be overstated. Given the table and flow-based nature of any ETL tool it will be at a minimum be very difficult and probably impossible for you to build a solution implementing Units-of-migration similar to the Business Objects of migFx.

Specification and implementation

In the case of an ETL based project, developers proficient in that particular ETL tool will be needed. Such developers may not necessarily possess the required business knowledge of the target and source system (or coming Source systems).

Building a new solution based on ETL, you risk encountering a problem common to basically all, larger data migration projects: The people with knowledge of data and systems must specify to the ETL developers how to migrate data. In the hand-off, the ETL developers do their best to implement the intentions of the specifications. The specifications will tend to grow indeed very large, and the ETL implementation will tend to drift away from the specifications as time goes by and the process iterates as the target system evolves.

In a one-off data migration, this setup may be acceptable. But in the case of repeated data migrations for years to come – as with your system – this approach is one reason why most migration solutions of this kind bog down over time and become increasingly expensive to maintain and enhance to match future evolvement of the target system.

The migFx approach is fundamentally different. migFx comes with a comprehensive *Studio* Windows application, where the people with knowledge of data and systems can collaborate and completely specify the migration. migFx automatically generates the implementation based on this specification. As part of the specification in Studio, it is possible to specify manual rules that must be implemented to supplement the generated code. This is the only manual intervention necessary that requires development skills (c#).

Developing and maintaining the migration in migFx means building the Business Objects and field mapping in Studio. This is done directly by the people with knowledge of the systems and data being migrated. The implementation is generated automatically. migFx requires a very limited number of c#-skilled developers to supplement the (relatively few) manual rules. It is our experience that 1 or 2 developers is enough for even large and complex migrations.

In contrast, developing and maintaining an ETL based solution will require substantial ETL-skilled development staff to develop and maintain a large heterogenous collection of ETL flows.

It is the foundation of Business Objects that enables migFx to generate the implementation in this way. Given the nature of ETL it is improbable that you will develop a similar solution where ETL flows are generated automatically. Even if so, you would still have to develop an equivalent to the migFx Studio application to provide the input to the automated generation of ETL-flows.

Standard and uniformity

As the need for ETL skilled resources illustrates, a migration solution built on SSIS will result in a significant number of ETL flows. These flows will be built by the ETL developers using the ETL design surface where data connections and transformations are dragged onto a canvas and connected with arrows designating the flowing data. Most - if not all - of these flows will over time become large and complex.

Though a set of standards will surely be imposed, it is inevitable that each of these flows to a degree will be bespoke in nature as they are manually developed and maintained. So even though ETL skills are readily available, the ETL solution will most probably be increasingly dependent on a limited group of key ETL developers that participated in the development and knows the ins, outs, peculiarities and many details of the solution.

In migFx everything is completely uniform and done in the same way regardless. All the Business Objects are mapped in Studio in the same way. The generated implementation will migrate all Business Objects in exactly the same manner. Understanding how to map one Business Object in Studio, you understand how any Business Object is mapped. Understanding how the migFx implementation migrates one Business Object, you understand how any Business Object is migrated.

There is no magic wand to make complex data migration simple. But migFx comes a long way of making it standardized and accessible, far less dependent on key people and far easier to bring new people on board.

Handling of errors

In an ETL-flow, many of the transformations in the flow will have an error output stream. Errors triggered by records in the input stream to the transformation will result in records sent to the

error stream. In a complex solution, with many ETL flows consisting of many transformations, there will be a very large number of such error streams. It will be a discipline in itself to ensure uniformity in order to consistently present this error information in a meaningful and aggregated manner. It will also be a challenge to provide an exact back-reference from any error produced from the migration to the exact point in which ETL flow the problem occurred

In migFx, it is an integrated part of mapping the Business Objects in Studio to specify where events should be raised and what variable information from the migration to include with the event. These events will be automatically emitted when migration executes in a consistent manner and presented to all stakeholders by the migFx Tracker application.

The Tracker application presents the events in an aggregated fashion ('this event was emitted by this number of Business Objects') and provides drill down to the single Business Objects emitting the event. The cross-reference capabilities in the Studio application makes it fast and simple to identify exactly where in the Business Object mapping a given event is raised and thus where to correct to eliminate the event.

With careful design from the outset, similar capabilities may be developed in an ETL based solution, though - given the lack of Business Objects - it is doubtful it will reach a level comparable to what migFx comes with out of the box.

Stakeholder involvement

An ETL based solution runs a significant risk of being technical and 'handheld'. In this case, the migration process will end up being a black box, accessible only to the ETL specialists with the detailed knowledge necessary to execute the migration and handle the many, complex ETL flows and their interdependencies. Errors, progress and the quality of the end-result will not be readily available to significant stakeholders, such as management and especially the responsible users of new customers in future migrations.

migFx is built with stakeholder involvement in mind. At the outset, reports from the migFx - Studio application provide valuable and structured information on the data requirements. During migration and testing, the migFx - Tracker gives a complete, logical and understandable view of the entire migration – both at an aggregated level and also in extreme detail to every single data field and every single error of every single Business Object.

The Tracker is readily accessible to management to give a clear indication of progress and quality. The Tracker is also available to the users involved in the acceptance test of the migrated data. The search capabilities of the Tracker allow the user to locate and view any Business Object and get a complete view of how the Business Object was migrated. The workflow capabilities of the Tracker allow test users and migration team to communicate clearly and concisely by commenting and assigning responsibility and status to the errors encountered during the migration.

Apart from being of great value during the lifetime of a migration project, the inclusion of stakeholders will also prove to be a valuable argument in the future, when negotiating with potential customers, that you have a robust and transparent process to safely onboard their data to your system.