Informatica

White Paper

Master Data Management and Data Migration

How to find data migration project success with master data management

About Informatica

Digital transformation changes expectations: better service, faster delivery, with less cost. Businesses must transform to stay relevant and data holds the answers.

As the world's leader in Enterprise Cloud Data Management, we're prepared to help you intelligently lead—in any sector, category or niche. Informatica provides you with the foresight to become more agile, realize new growth opportunities or create new inventions. With 100% focus on everything data, we offer the versatility needed to succeed.

We invite you to explore all that Informatica has to offer—and unleash the power of data to drive your next intelligent disruption.

Table of Contents

The goal: Bring the relevant data into a consistent structure and eliminate duplication across and within an organization where MDM serves as the single view of truth after migration is completed.

This paper describes the significance of master data management for data migration projects, its usefulness, and the best deployment options, including relevant case studies.

Why MDM Is So Critical to Data Migration

Every new system needs data to prime the pump of activity. Most new systems require an awful lot of data. Today, we're migrating data to new systems from a multitude of legacy sources in addition to enriching new systems from external sources like Post Office address files. The Informatica® data migration tool kit deploys a suite of technologies and best practice processes to solve for the range of challenges arising from today's data migration scenarios.
First though, let's set the scene. Though our target system may be all-encompassing, our data sources are just as likely to be stove-pipe solutions, each one designed around a different business process with a different worldview. Though the target system will demand consistency, the legacy environment often is inconsistent in structure and content. This creates two issues for us; the use of MDM techniques is appropriate in both cases.

Issue #1: Getting an Apples-to-Apples Comparison

Let's look at an example where we're installing a new production planning application. It promises to enhance shop floor management and increase efficiency by linking our accounting and human resource applications. However, we find that each of these departments, and the systems that run them, have a different take on exactly the same real- world things. The world seen by an accountant is one of cost centers, profit centers, depreciation, capital assets, and operational assets. But the production engineer defines the same physical space by the following attributes: automated, semi-automated, and manual processes; workflows; maintenance schedules; and production quotas. And the HR department sees the same scene as a question of internal staff, external staff, training requirements, skill levels, pay agreements, and health care issues.
They look at the same things—but see them differently.

Consequently, it's not surprising that when they choose and design their systems, they model them quite differently. It's not that any of them are wrong; however, they are inherently inconsistent. As a result, when it's time to perform our data migration, we really do find that we're not comparing apples to apples. Rather, we're comparing apples to pears. To be successful, we only want one type of fruit.

It's not as if we can just dismiss one or the other view either. None of these multiple truths is "wrong" in an absolute sense. They're just different views of the truth. So, even if we decide that the production department view is the most apposite (given that is our driver for change in this case), we can't insist that the other legacy data stores be modified to fit the production department's model without re-engineering them on a scale potentially equal to the migration itself.

It's more likely we take a view that each legacy data store is right to the extent that it models its own domain. Meaning, the production department is right from a production perspective, the HR from an HR, and so on. Therefore, we need a model that can accommodate all views. Interestingly, this challenge is exactly the same as the project to deliver the target system. Why not just wait for the target to be ready and perform our gap analysis against that?

On-Time Launch: An Intermediate Step

If the first issue is reconciling a fragmented data landscape, let's also take a step back for a moment and think about the timeline of a normal migration.If we assume the design and installation of the new system takes a year, then the target system will not be available until the eighth month at the earliest, and then only in a first-cut form. (And that assumes a standard project methodology, including project start-up activities, current-state analysis, new system configuration and business process re-engineering, etc.). Given our experience, we would expect stable target delivery in the tenth month—and even then, there'll be changes right up to cut over.

Based on our experience, we may have as little as two months to perform our gap analysis, work out our extracts, transformations and loads, write and test our load scripts, etc. Unfortunately, this isn't nearly enough time to perform a high-quality analysis of the structural differences between our legacy data stores, let alone produce the composite data items we need to have in our load files.

One solution to this timing issue is to create an intermediary model (what we call a migration prototype). We record the differences between each legacy data store and the prototype and begin our clean-up and data preparation activities. Then, when the target is finally delivered, we look at the differences between the prototype and the target. Because we already know the transformations between the legacy data stores and the prototype, we can isolate the differences between our prototype and the eventual target, thus greatly simplifying our activity in the busiest, most highly-pressured time.

This is a neat trick: Build a migration model that's a best guess as to what the target will look like, analyze the differences between legacy and migration model, and work out transformations, data enrichments, enhancements, etc. Then, when the real target appears, we tweak these transformations with finer granularity issues that come out at the end. This way, if we get our prototype only 80% right (and experience suggests that we'll have gotten it a lot closer than that), we will have designed 80% of our transformation logic before the challenging period of the last few weeks of the project are upon us.

But why is a Master Data Management solution going to help as part of this model?
According to Wikipedia: "MDM has the objective of providing processes for collecting, aggregating, matching, consolidating, quality-assuring, persisting, and distributing such data throughout an organization to ensure consistency and control in the ongoing maintenance and application use of this information"

From a data perspective, the three domain views from our example—the accountant, the production engineer, and the HR pro—will be instantiated in coded values within relevant data stores. So, we're very much interested in providing a process for "collecting, aggregating, matching, consolidating, [and] quality- assuring".

For the sake of our project, we're not so directly concerned with "persisting and distributing such data throughout an organization" for the reasons we have discussed. But ensuring "consistency and control in the ongoing maintenance and application use of this information" is a primary concern for our migration project. In other words, we take an MDM approach to data migration, except—because our job is to replace not enhance systems—we're unlikely to close the loop back to the systems that use these data items.

Case Study: A Zillion Products

But let's look at a real-life example to make all this a lot clearer.

The data migration in question was taking place at a large telecommunications company. To give an idea of scale, they numbered their customers in tens of millions and the number of installations in hundreds of millions. The number of discrete products was also large—up to a hundred thousand individual products. And just to make the whole thing more interesting, telecoms build products on products—meaning we had their version of the parts explosion to deal with.

Telecommunication companies are also challenging because of the number of legacy systems they have. Because ordering, designing, providing, and billing range over multiple layers of physical and logical activities, they have a tradition of building point solutions for each step of the way. The result is that out of a potential pool of 400+ legacy systems, we're generally dealing with around thirty for each installation.

Each of these legacies, of course, had its own view of the world related to where it was in the process flow and if its view was logical, physical, or financial.

For the purposes of this paper, we'll look just at the problem of the structure of the products and why an MDM-based solution was imperative.

With 100,000 products, some of which were built of other products, there were an awful lot of rules. Where one legacy applications saw a single phone installation, another system saw a handset, a dial tone, outward call dialing, inward call barring, ring back, answer phone, etc. In other words, there was up to a dozen products just for a simple house phone, never mind a complex voice and data network. It seemed like every system in the provisioning process saw the phone differently. We needed consistency, and we needed to get started on that as soon as possible. Waiting for the target was just not an option.

The solution was to create a Master Data Management hub for products and product builds. A weekly refresh addressed the fast-moving industry requirements, where new products are designed and added all the time. The hub allowed us to check legacy data stores for matching products as well as maintain a view into the differences between representations at different points on the delivery chain. As each phase of the migration occurred, we knew the differences between the structures of our source systems and the target and could recode our mappings and transformations accordingly.

Issue #2: Quality Matters

Now let's look at the second type of issue—the content or data value issue. Just as our legacy data stores can have multiple structures for the same business object, we can also have multiple values. Let's have a look at a few common examples of this.

A bugbear of all marketing departments is the duplicates contained in the customer lists gleaned from the multiple data stores that make up their hinterland. Some will be easy to spot and eliminate. Others fall foul of the dreaded homonym or synonym problem. Is John Smith the same as J Smith, who may be J P Smith, or even Jonnie Smith elsewhere? When we perform a migration, we all have the same problems. These are just as likely to occur within the same data store where multiple copies of the same person can be present.

These problems are often compounded by structural issues. For instance, in business-to-business (B2B) environments, there's often confusion as to who the customer is. Should we consider the customer to be the top legal entity? (That is, the corporation we would haul into court if it came to it.) Or is the customer the trading arm we deal with? Or a local depot or store? Again, your view on this will change depending on where you sit in the business. The logistics team will look at points of supply; the billing team will need to know the points of supply and the billing address details. You may have sales teams that operate geographically but with a layer of strategic relationship managers for larger clients. So legally, there may be only one legal entity, but there may be hundreds of local order, delivery, and billing points.

Once again, it may not be feasible to correct these anomalies in the source systems, especially if the source systems are not wrong. (Of course, there will be straightforward duplicates caused by faulty processes where the same customer has been created twice. These are wrong even in the terms of the legacy data stores and can be corrected within them).

So, where to start and how will an MDM solution help? It goes without saying that MDM is the perfect solution for mastering our key entities. Here, we'll be carrying out nearly all of the functions of a fully-fledged MDM including ensuring "consistency and control in the ongoing maintenance and application use"—at least to the extent that it relates to the duplicates that should not be in our legacy data set.

Case Study: Bringing Together Corporate and Local Systems

Another real-world example will clarify the challenge. A mid-sized banking organization needed a migration. Their offices were regionally dispersed, and within each office, there was a discrete instance of the corporate systems, linked together by a series of batch processes to head office. We had both sets of problems. There were local variations around structural interpretations and there were duplications within and across branches.

With the aid of MDM technology, we could build a single view of the truth and link back to the relevant source systems. The goal: Bring the relevant data into a consistent structure and eliminate duplication across branches and within branches at go-live time. In this case, the MDM went on to serve as the corporation's single view of truth after the migration completed.

Data Migration as the Gateway to High-Value MDM

The persistence of MDM is one of the key benefits of using the compelling event of a significant data migration as the springboard to realizing your MDM. In the course of preparing data for the migration, we'll need to de-duplicate and resolve the semantic issues around structural differences.

In the past, we've often created a throwaway application to hold a single view of the truth as long as the project needed it. If we use an industrial- strength MDM solution, the logic of carrying this work forward to complete the build of an MDM and its full integration into the fabric of your architecture is so much more compelling. You'll already have shown the value of MDM technology. And you'll be familiar with both the technology and the associated semantic issues that surround MDM adoption.

Worldwide Headquarters 2100 Seaport Blvd., Redwood City, CA 94063, USA Phone: 650.385.5000, Toll-free in the US: 1.800.653.3871

IN09_1120_02752

© Copyright Informatica LLC 2020. Informatica and the Informatica logo are trademarks or registered trademarks of Informatica LLC in the United States and other countries. A current list of Informatica trademarks is available on the web at https://www.informatica.com/trademarks.html. Other company and product names may be trade names or trademarks of their respective owners. The information in this documentation is subject to change without notice and provided "AS IS" without warranty of any kind, express or implied.