

# **White Paper**

## **Data Migration**

**iergo<sup>®</sup>**



## **Data Migration**

### **Introduction**

This paper discusses the inadequacies of a traditional techno-centric approach to Data Migration, examines the common pitfalls and suggests an approach that greatly increases the chances of a successful outcome.

First though let us define the nature of the Data Migrations we are discussing here.



## Data Migration

### Reasons to Migrate Data

Data migration is the movement of data from one system (or more commonly many systems) to a new repository.

The systems from which data is moved are commonly referred to as Legacy Systems. This makes us feel better when discussing their inadequacies – we didn't build them we just inherited them!

The system we are moving data to is commonly known as the Target System.

Others have pointed out the limits of this nomenclature but for the purpose of this discussion we will keep with these familiar titles.

Of course there are many reasons why data may need to be moved from one system to another including:

- System upgrades
- Business Process Re-engineering (BPR)
- New business processes
- Technology upgrade
- Company mergers and de-mergers
- Creation of reporting repositories (this is an example that makes the term Data Migration erroneous – the data is not being permanently moved from one repository to another, it is being replicated)

In this paper we are not concerned with simple version upgrades to existing systems. For these there are often industry standard tools and methods proved in practice. This paper is concerned with the essential "one off" move of data from Legacy system(s) to Target system where there is no established path or single tool.



## Data Migration

# Typical Migration Issues

Now let us look at the typical issues that face a migration project.

### **Plethora of Legacy Systems**

In the anarchy of the post Client Server revolution, every company desktop is a potential computer centre in its own right. Systems proliferate be they spreadsheets or fully-fledged, departmentally purchased databases. When we are tasked with migration, we are often faced with a myriad of possible legacy system combinations. And because these systems were developed by local initiative, there is no overall architectural structure documented for us to use as a guide.

### **Lack of Documentation**

The first issue leads into the second and as we shall see the third. Given that these systems are often developed by non-IT professionals (and even, unfortunately, when they were developed by an IT professional) there is often no documentation to accompany them. Their design, functions and sometimes even their existence, is a local secret.

### **Lack of Referential Integrity**

Again because these systems were developed as local solutions to local problems, their ability to communicate across departmental boundaries is often limited. They were never intended for that purpose. When we come to access them for data migration we find that common points of reference are missing.

### **Legacy/Target Data Structure Miss-Match**

Often, even with what is on the surface a system upgrade or like for like replacement there is an element of BPR. Two systems performing the same business function may be subtly (or not so subtly) different in data structure. Even an issue of granularity between two similar systems can cause problems. The data structures simply do not match. This can lead to awkward migration decisions, often concerning historical transaction data, never mind in flight transaction data.

### **Missing Data**

Often, in the middle of a migration exercise we are left scratching our heads wondering how the business survived on the data it apparently uses. Whole parts of it are missing. This is ignoring the data gaps highlighted above that relate to structural differences between different systems. It sometimes seems that data that logically had to exist for the company to function just is not there.

### **Known and Unknown Data Errors**

Often when we initiate a discussion about migration with our user population we are met with a wry smile and a frank admission that the data in System X has known defects. Worse still, it is only when we come to load time that data errors become apparent. The business has developed all sorts of work-around's to cope with data that is either missing or just plain wrong. Yet the business case and the management expectation is that the Target system will have perfect quality



## Data Migration

data. And there is never enough time at the end of a long project to correct errors that have withstood correction, in some cases for years.

### Personnel Issues

The most difficult – even to describe. We frequently encounter resentment, often bordering on open hostility to us in our role. Sometimes this is due to the last migration or system initiative and the problems it caused, sometimes it is grounded in the understandable fear of losing control over data that the local experts have spent a considerable effort to polish to a state that they can then work with. How do we get the business to work with us and not against us?

### To Recap

There a number of issues that in combination make what may seem, at the outset, a simple problem into a very messy problem indeed. For the purposes of the remainder of this paper I'm going to consider them under three distinct headings, although each, when encountered on real projects has its own unique characteristics and most could be placed under more than one heading.

- Data Quality Issues
  - Lack of referential integrity
  - Missing data
  - Known and unknown data errors
- Data Structure Issues
  - Plethora of Legacy systems
  - Lack of documentation
  - Legacy/Target data structure miss-match
- Personnel issues
  - Previous bad experiences
  - Ownership and control
  - Politics



## Data Migration

### The Traditional Techno-centric Approach

The traditional approach to Data Migration is to see it as a technical issue. We characterise it as a problem of moving bits and bytes around. We focus on issues of storage size, runtime length, back ups and extract, transform and load (ETL) coding.

But how well does this deal with the problems above?

#### **Techno-centricity and Data Quality Issues**

Here we seem on the firmest ground. From our first days at programming school we have been taught about data validation. We know about range checking, value checking, reasonableness checking, referential integrity checks and data type checks. We can parse data strings and check for message length constraints.

Data Quality even sounds like a technical criterion.

And yet when we hit a problem how do we fix it? Typically we don't. We throw out the errant data item and carry on with the ones that are "good". But where do the ones that are poor go? Usually they go straight back to the business.

But what is the business to do with this flood of "errors"? What preparation and training have we given them to be able to sort out these "errors"? They typically have no coding skills. The term "Referential Integrity" has no meaning to them. In the time pressures of the day job (and it is the day job that is paying all our wages) they have no time to hand correct possibly thousands of data items. There is often no budget for them to carry out this function.

And look at the situation from their eyes. In the case of known data errors, well these may have been with the company for some time. If they couldn't fix them then why do we believe that they can fix them now? There can even be an element of gloating at the misfortune of the Technologists. Didn't they believe that they could sort out the problems with the data?

A gap begins to develop between the project and the business it is supposed to be serving.

In the case of unknown data errors this is both a political and a business case challenge. On a political level, handled with the usual insensitivity of the average technologist, this can be perceived as a personal criticism of the competency of management and staff. The IT department has assumed the role of policeman, the business is the accused.

The gap between Business and the project starts to widen.

From a business case viewpoint the business may well ask, defensively, why do these data errors matter? The business has been successfully conducted before, why is it important now? It only seems to matter to the technologists.



## Data Migration

Now it is the turn of the sponsoring business managers to draw back from wholehearted support of the proposed new system solution.

None of this of course addresses the bigger issue of where the validation rules that the technologists are applying come from.

We can do pattern matching in existing data for instance. This will tell us something about the shape of current data. It will not tell us if that shape is right.

We can assume from data names that, say, an address should be of Post Office Address File Format (PAFF). It does not tell us that this company has non-post office addressable locations to deal with (a common problem for utilities and construction companies).

And most obviously of all, we cannot tell how far the data we have represents the existing business reality. Are these Suppliers still trading with us? Are these two customers with similar names really the same people?

### Techno-centricity and Data Structure Issues

Again this looks like an area where the technologist should be in their element. With the expansion of the capability of Extract Transform and Load (ETL) tools, the technologists should be in a position to parse, consolidate, reformat and apply the data that is given them (allowing of course for the Data Quality constraints above).

Would that real life were that simple.

The plethora of legacy systems is the first stumbling block. How to choose? The commonest response (and probably the most alienating and arrogant from a front line workers point of view) is to insist that data will only be extracted from a small set of enterprise systems. Normally those systems supported by central IT. This has obvious attractions to the tidy mind of the Technologist. It removes the complexity of choice, it gives the technologist what they want – access to data of a documented format, with known Application Program Interfaces (API) that their extract technology can work with, it seems more disciplined and it shifts the onus of tidy up work from them to the business.

But this ignores a couple of key issues. Why did the business find it necessary to build their own data stores? Are we after the best quality data we can get or are we just going for that which is most accessible to our technology?

Isolated data stores are built for many reasons. Sometimes it is just more convenient to store data for local reporting or management reasons. Sometimes the data in the enterprise system is awkwardly structured for the task the business is trying to perform. But sometimes the data they need to do their jobs is of poor quality, or missing in the corporate system.

When we, on the project, find a data gap, are we certain that there is not an overlooked data store that might hold this information?



## Data Migration

If we want to further alienate our fellow workers then ignoring their cherished and hand crafted data stores, that contain data missing from the corporate systems, is a pretty good way to go about it. Often the local data store is the embodiment of the knowledge of key business domain experts, yet we dismiss them because they fail to comply with our business domain demands of referential integrity, external consistency, normalisation and build quality. But what is our end goal here? We legitimately want to provide our customers with the technically best solution we can, but don't we also want it stocked with the best quality data we can access?

Of course even if we are in agreement with the need to gather data from the best source not just the easiest we wont find the answer to what is the best source in the ETL tools, only that which complies with some fairly arbitrary validation rules.

One of the things that trips us up here is the lack of documentation of these local data stores. But then some of our target legacy systems may not be that well documented. At least local data stores tend to be less complex and the author is often at hand to explain how they work.

Surely though Data Structure Miss-matches must be within the compass of us technologists? Well yes and no, but of course it is the "No" that catches us out.

Expressed simply, there are mapping problems that are not amenable to logical solution without access to external information.

The most obvious is the many to many problem where an item in list A can be mapped to more than one item in list B, but each item in list B can also be mapped to more than one item in list A.

There are also other more devious problems. There is, for instance, what I call the one way street problem, where, much like the one way encryption used for passwords, it is possible to navigate from A to B but not possible to go back again. If we need to go back to discover some future relationship for a data item it is not possible. Because the reverse journey was never envisaged as necessary when the Legacy system was designed, there are no history values held that will help us. This kind of problem occurs, most frequently, when we try to migrate historical transactions. There are also problems of new systems requiring increased granularity.

So all in all we will often find ourselves having to go back to the business for answers that only the business domain experts can provide, possibly by reference to those local data stores we were so contemptuous of.

### **Techno-centricity and Personnel Issues**

So we've been humbled by our lack of business domain knowledge, thwarted by data gaps that we could not bridge, stopped in our tracks by data structural issues and perplexed by data quality issues that we did not expect. We now turn to our colleagues in the business for help.

We find them strangely uncooperative. Disregarding the fact that we have cascaded an unforeseen avalanche of erroneous data over them, publicly humiliated them over the quality of their data, arrogantly ignored their specialist



## Data Migration

knowledge both of the business domain and of the data quality of existing systems and contemptuously dismissed the local data stores that they know contains the best quality specialist data, we are still surprised when they seem hostile!

Oh yes and of course by this time, in many migration projects, the deadline for migration is looming close and failure is a real possibility. Why are we surprised that we encounter reluctance to get involved?

We have implicitly taken ownership of the data from the beginning of the project and now we want to give it back. Our characteristic insensitivity and hubris has done nothing to bridge any gaps that previous system implementations have created and if there are any project assassins out there we have only provided them with more ammunition and a bigger target to hit.

This is the point when many migration projects, and with them the reputation of the larger programmes, spin down a vortex of mutual recriminations and frustrated expectation.

It is no surprise that many experienced and otherwise excellent system implementers steer clear of taking on responsibility for Data Migration.



## Data Migration

### So Is There a Better Way?

The simple answer "Yes", but this is a complex and specialised area. If you are planning a programme that will involve a significant migration exercise, get expert advice. There are techniques and project structures that only consultants who specialise in this area will know. And they are best placed to advise on how and when to use them.

Do not allow that advice to be biased by being slanted towards one technology or another.

Import expertise for specific technologies is necessary. It is necessary but it is not sufficient. Knowing how data can be best imported into the Target system is useful at system go live, it is not of so much assistance in the 7/8ths of the project that precedes it where data selection, extraction and cleansing are the key.

ETL and data profiling tools can be huge time and money savers but only when used within a coherent strategy. And that strategy has to be right for your migration needs, in your company, at this point in time. There is no one size fits all solution. Each migration is different and a migration strategy should be tailored for it.



## Data Migration

### Still Want to Go It Alone?

First of all good luck! Data Migration can be a very fulfilling activity that will build lasting bridges with the business community and greatly increase your understanding of the real issues that face your co-workers.

Data Migration can also be the disaster that sinks the programme and the careers of those involved. In exceptional cases it can also sink the enterprise.

How do you put yourself in the first category and not the second?

#### **Plan Early**

Analyse the environment you find yourself in and create a migration strategy that best suits your situation. Your migration strategy should consider each of the issues outlined above and provide solutions to them that will fit with your enterprise's culture and corporate strategy. Your Migration Strategy should be prepared at programme initiation. Any later and you risk decisions and assumptions having been made that will dog your work for the whole duration of the programme.

#### **Plan Big**

Data migration can be complex even when you are moving small amounts of data from one spreadsheet to another. Its complexity grows exponentially with the number of systems being migrated and the number of columns of data. Assume that you will need significant resource from the business as well as the programme, to identify, prepare and transform legacy data.

#### **Plan Loose**

Expect that there will be unforeseeable problems along the way. Make space in your plan for the likely iterations that will be forced upon the less experienced practitioner.

#### **Be Business Facing**

As I have shown, a techno-centric approach to data migration is almost always going to be a struggle that risks alienating the business colleagues you are supposed to be serving. Your strategy must be explicit about how you will retain business ownership over migration and the migration products. The business managers that will be called upon to commit resource must endorse it.

#### **Stick to the Plan**

Do not allow yourself to be derailed by special interests (especially those of the technology suppliers).

Do not look for automated tools before you know how these tools will fit into your strategy. In conversations with potential suppliers, see how their thinking will fit in with yours. This does not mean that you cannot amend some of the lower products so that there is a better product fit, but do not abandon your business facing approach.



## Data Migration

### Have Fun

Building a virtual team of Technologists and business people can be hard work but also very rewarding. The Technologists are often even more reluctant than the enterprise to leave the comfort zone of their normal roles. Data Migration is often seen as lacking in technical challenge but big in career risk. On the other side the enterprise may be wary and suspicious of yet another initiative from the centre. You will need all your interpersonal, diplomatic and political skills to overcome these difficulties.

Once again good luck and if you have found any of the forgoing of use please contact me at:

[john.morris@iergo.co.uk](mailto:john.morris@iergo.co.uk)