White Paper

Bloor

Next steps for Data Integration

A White Paper by Bloor Research
Author : Philip Howard
Publish date : June 2012

...this approach requires far less in the way of resources, particularly with respect to personnel but also hardware and software resources, and therefore costs, than traditional methods of supporting data integration

Philip Howard

Introduction

This paper examines the issues facing data integration tools today and in the near future. We consider why these issues represent significant challenges and what can be done to face them and, indeed, if anything can be done to meet these challenges. In particular, we will discuss what we believe to be the fundamental problem that makes these challenges so daunting and we will consider whether there is a better way of doing things.

Today's solutions and challenges

The fundamental issues facing all data integration tools or, more specifically, what used to be referred to as ETL (extract, transform and load) tools are that they are too complex, don't perform, won't scale, take time to deploy, require a significant skill level, are not easy to use and are too expensive. These are the basic reasons why something between 50% and 70% of all data integration tasks are hand-coded by users that eschew the use of data integration tools. We will examine each of these characteristics in turn.

However, before doing so it is worth noting that the buyers of software (particularly in large enterprises) are conservative by nature and the major software vendors tend to reflect this by maintaining the product status quo for as long as possible, often at considerable cost to the end user community. For example, analyst Dennis Gaughan recently spoke at a symposium in Australia. According to IT News he gave a broad overview of the strategic direction of the world's largest application vendors: IBM, Microsoft, Oracle and SAP, and had some strong recommendations as to how CIOs should engage with each of them. Gaughan said none of the four vendors are "re-imagining" IT. "*You won't find innovation in their product portfolio*," he said. "*You might find it if you try and talk to the research parts of these organisations. I would argue that a lot of what they are trying to do is keep the status quo and find ways to increase share of wallet. There isn't an innovation agenda. They have to think about (their company's) investors. If they get on the leading edge, it exposes and impacts them on the short-term*."

In a nutshell, the current state of the data integration market exactly reflects this lack of an 'innovation agenda'. We would argue that, contrary to the maintenance of the status quo, companies willing to break ranks on innovation will break away from the pack. How to do so forms a part of this paper.

Over-complexity

Data integration products are essentially developer's tools. Steps have been taken by vendors to make data integration more susceptible to use by business analysts, such as the introduction of business glossaries and simple diagramming capabilities that can be mapped into a developer's specification, but there are still multiple steps to perform: extracting metadata from the source or sources, mapping it to the metadata extracted from the target(s), defining datatype conversions, defining transformation rules and so on. Rules, in particular, can not only be very complex but they can grow exponentially. We recently encountered one company that had 37,000 transformation rules. The entire department spent all of its time simply maintaining these rules and had no time for new development. Of course, this may in part be to do with a lack of reuse but while suppliers encourage reuse they can do little to ensure that it happens in practice.

The basic problem is that there is a lack of automation. Without much more automation, data integration will never be a tool for the masses. Certainly there are pre-built transformation functions provided but you still have to tell the software how to perform relevant transformations. In other words the environment is essentially procedural (though things like parallelism are built-in) rather than declarative. Let us be specific:

- It would seem reasonable to be able to use a semantic approach that could recognise that 'Cust_no' is the same as 'CustID' in two different data sources. You could therefore provide a customisable semantic dictionary that would automate, to a large extent, the recognition of equivalent columns. expressor Software actually did this when it first released its eponymous product. What it found was that its customers had many tables and columns that had obscure names that had no obvious correlation with their content. While a few of its customers used the facilities (since dropped) by expressor to build a complete semantic dictionary, most did not, purely because of the work involved. Of course, if all IT departments had sensible naming conventions that were adhered to rigorously then this wouldn't be an issue: but they don't and it is.

Today's solutions and challenges

- Where a semantic approach does work well is with respect to automation of datatype conversions. Where you want to convert a 30 length character string into one of 50 characters or, more interestingly, where you want to convert feet into inches. Such functions can and should be automated, though that is frequently not the case. Of course you can define a rule that does these conversions for you (as you could for the conversions in the previous paragraph) but the whole point of automation is to make such rules unnecessary.

- You need to be able to automate the recognition of target datatype constraints. For example, Oracle used not to (and may still not) support the Small Integer datatype. If you have sources that use this datatype then you would really like your data integration tool to tell you that you will have to perform a datatype conversion before attempting to load the data into an Oracle database. Otherwise the automation referred to in the previous paragraph will break down. Of course, you could manually define this in order to support conversion but that would defeat the point. Unfortunately, reading the metadata won't help here: it will tell you the datatypes in use but not what's permitted, so the only way you can do this with conventional tools is to hold detailed reference tables for every version of every database that is supported, which, of course, nobody does.

These examples are not intended to be exhaustive. There are, no doubt, many other areas that could be automated (indeed, there are some that pertain to performance—see next) but the point should be clear: traditional data integration tools are much more complicated than they need to be.

Another point about complexity is that data integration tools were originally designed to do a single job. That is, they were developed to support the batch-based movement of data from transactional systems into a data warehouse. However, over the years vendors have added further data integration capabilities to these basic ETL capabilities until data integration 'platforms' are now capable of multiple functions—not just ETL and its derivatives but also supporting change data capture, B2B capabilities, EAI, data migration, archival, master data management and so forth. While there are certainly overlapping areas of functionality in all of these spaces they also have separate requirements, which means that these platforms have had more and more complexity added to them. Moreover, such complexity typically means that, in effect, you have multiple tools that share a common metadata repository. Although suppliers typically describe such environments as 'integrated' that is frequently not the case in the true sense of word.

People issues

Because of their complexity, mainstream data integration tools are, effectively, developer tools. Learning to use a new data integration tool is effectively like learning a new development language (or perhaps a 4GL): it takes a significant effort and significant training is required. That means implementation times are extended and time to value is lengthy. Further, while some efforts have been made by some vendors to make data integration more amenable to use by business analysts, the truth is that these are, at best, only a partial success.

Today's solutions and challenges

Performance

The earliest data integration tools used compiled code that ran on the target system in order to maximise performance. Unfortunately, this is an inflexible solution and, besides, it ties up the target system when you really wanted it to be doing other things, so the dominant approach to data integration rapidly became ETL, whereby an intermediate server was used to host transformation processes and relieve processing from both the source and the target.

By the early part of this century this approach was starting to see this intermediate server become a bottleneck and companies like Sunopsis (now part of Oracle) came to market with an ELT-based approach. This uses the MPP-based nature of the most common targets (data warehouses) to perform transformations. However, this is not useful if the target is not a warehouse or isn't an MPP-based warehouse. Moreover, with batch windows shrinking and more and more companies wanting real-time or near-real time (micro-batch) loading this means that transformation processing simply puts an additional load on the warehouse precisely when you don't want it. As a result, many leading vendors now offer what we might call 'transform anywhere' capabilities; on the source, in the intermediate server or on the target, or any combination of these, as appropriate.

As far as where you actually perform transformations is concerned, there is no farther to go (but see below). This means that either those transformations need to be more efficient or you need to throw more hardware at the problem. The latter appears to be the preferred option for most vendors: have a bigger server or more nodes in your cluster and you can increase your performance. Unfortunately, that's not likely to be the preferred option for users; more tin means more cost and more administration. Of course, an alternative would be for greater support of in-core parallelism but that would mean re-writing the software.

What should occur (though we have seen very little evidence of it) is that suppliers will build more optimisation into their products. They will, in effect, introduce the equivalent of a database optimiser. This will effectively have three functions to perform.

1. It will support the transform anywhere capabilities already discussed. For example, IBM already supports both ELT and ETL processes for Netezza and you can choose which option to use and you can dynamically amend that choice by, say, time of day. However, the problem is that you have to define even more rules about when you do this: for example, "between 8.00am and 5.00pm use an ETL-based approach but otherwise use ELT". A much simpler option (for the user) would be if the data integration product had its own optimiser, collected statistics and so forth, just like a database does, generated an 'integration plan' (as opposed to a query plan) and generally automated the whole process of where you perform relevant tasks.

2. Once such an optimiser is available the next thought would be for it to be able to re-write scripts, just as database optimisers re-write poorly written SQL. If the developer is joining data in a less than efficient manner or if he is sorting and re-sorting the same data or doing anything else that is less than optimal then the optimiser would take care of this and thereby improve performance.

3. Another performance issue with many approaches to data integration is that transformations are performed on a field-by-field basis. That is, you access the source data, transform it and write it to disk. This is not very efficient from an I/O perspective. From a performance perspective you want to minimize the number of I/Os at both source and target, so a smart optimiser with read-ahead capability should be able to pre-fetch data in an optimal fashion. Of course, if you could simply read all the data into memory that would be simpler still, and faster.

Apart from where you actually perform transformations and how you access the data, the other potential bottleneck for traditional data integration solutions is the requirement for a metadata repository. Now, such repositories have a number of advantages; they enable integration between tools, they support business glossaries, and so on and so forth. However, they also add complexity to the whole environment and they represent a performance issue, because you have to keep accessing the database in which the metadata is stored, which potentially means I/O problems. The natural

Today's solutions and challenges

solution would be to hold metadata in-memory (perhaps in an in-memory database) or on solid state disks but these are, again, relatively expensive. Moreover, most data integration products are extremely profligate with their use of memory to the extent that the servers they run on are typically incapable of running any other tasks. There are exceptions. Syncsort, for example, is parsimonious in its use of memory, mainly because it was originally a mainframe product that was first developed when memory capacities were very small.

None of this bodes well for current products in the market. There are remedies for their performance constraints but these involve significant costs either on the part of the supplier (re-write the code, develop an optimiser and so forth) or the user (buy more tin) or both.

Scalability

Scalability is really the flip-side of performance, although scalability is technically about the number of jobs that can be running with appropriate performance rather than just how fast any particular task is. Needless to say, if you improve performance then you should improve scalability. The one idea we have not discussed, and which we believe is likely to appear in the market (though perhaps not from any of the leading vendors) is a Hadoop-like approach to data integration. That is, a large cluster of cheap servers running MapReduce to parallelise ETL jobs. Given the likely increase in requirements for loading 'big data' for analysis purposes it seems inevitable that there will be a move in this direction.

Costs

The more hardware you need to get the performance and scalability you require, then the more any solution will cost. Then there are license fees and support costs and the more sophisticated the solution, the more these fees will be. On top of that are the internal costs involved in developing, maintaining, using and administering the chosen solution, which, in our view, are likely to exceed all the other costs put together. It is for this reason that a majority of data integration projects are hand-coded: because the tools in the market today are too complex and, in particular, they do not offer enough automation.

IP Issues

The development of mainstream data integration tools pre-dates many of the technology advances of the last two decades such as XML, XPath, Xquery and so on. Given the relatively recent demand for XML functionality in data integration, the software industry has attempted to 'bolt on' XML functionality rather than embedding XML and related technologies as architectural foundation components within an integrated platform. This approach can add considerable complexity, cost, and performance constraints, not to mention the potential exposure to Intellectual Property issues with early adopters of XML-based methods. This opens the opportunity for smaller and nimbler software developers and new entrants to gain first mover advantage by quickly exploiting new technologies, designing new methods of using those technologies, and patenting those designs and methods. This has led to a robust market for software patents as user demands drive the major vendors to introduce more cost effective integrated solutions.

While patent infringement litigation has become a nuisance for mainstream software vendors, they have profited enormously from Intellectual Property protection. What is more important is that the slow adoption of new technologies has led to the lack of integration discussed above and resulted in complexity and cost to users, arguably costing them many billions of dollars of user benefits each year, just in data integration alone.

What is needed as we move into the future?

What is going to happen in the future is more of the same. In fact, as we enter the era of big data, there will be a very great deal more of the same and with batch windows shortening, or disappearing completely, the issues discussed above are only going to get more acute. In particular, there will be an increased emphasis on the ability to scale the integration solution, either on scale-up or scale-out basis, or both. However, an explosion in the quantity of data to be moved is not the only thing that's going to be happening.

The vast majority of the big data that companies are going to want to analyse is driven by the Internet. Not only is this growing it is also changing: the 'semantic web', otherwise known as Web 3.0, is around the corner. As this is adopted this will change the nature of the issues facing data integration vendors. In particular, the semantic web is semi-structured rather than unstructured. Specifically, web-based information will have metadata associated with it, which it does not today. This metadata will be captured in RDF/XML format so there will be an onus on data integration vendors to capture that metadata in order to support relevant transformation processes.

Now, data integration vendors all support the movement of XML-based data but they either do it by treating it as unstructured data (a document) with no metadata, or by extracting the metadata from the self-describing details within the XML document and effectively treating the information therein as if it was relational data. This does not seem a very sensible approach as we move into the semantic web: if 70 or 80% or more of the data to be analysed is in XML format it would seem to make sense to have specialised facilities for supporting XML-based formats rather than trying to fit them into other pre-existing approaches.

However, as we shall see, we believe that this growth in the use of XML-based data does not so much pose a problem as represent an opportunity.

The semantic web is not the only thing we are going to see more of. We are already seeing more and more cloud-based deployments and the use of Software as a Service (SaaS) based solutions is on the uptake. Both of these environments pose further data integration challenges because of the need to transfer data between on-site and off-site locations.

More generally, there is clear demand to make data integration environments simpler, easier to deploy and usable by business analysts without need for reliance on developers: if you want to transfer data to and from salesforce.com, for example, you really do not want to have to produce a specification, explain it to the developer, have him develop it only to misinterpret it and then go through that whole cycle again until it is right. Of course, some tools allow business analysts to create a high level specification, using a graphical interface that can then be passed to the data integration specialist for development. However, frankly, users don't even want to do that: they want press a button and have it happen. Until companies can provide that, at least for simple requirements, users will continue to roll their own.

Tralee's XML solution

We have considered the various issues facing data integration suppliers both now and moving into the future but what is behind these problems?

In our view it is metadata that is at the heart of the issue. Of course, you need to understand metadata in order to support data integration processes but the way that that understanding is accomplished is unwieldy and slow. Consider: you have to extract metadata from both sources and targets, you have to store in it a repository (which is potentially I/O bound), you have to reference it all the time, and you have to administer and maintain it. If you could capture metadata implicitly rather than explicitly, and if you did not have to store it, then that would greatly simplify the process of data integration.

Such a method exists. XML documents are self-describing. That is, they contain their own metadata. With XML you do not have to extract any metadata, it is already there. Nor do you have to store that metadata anywhere because it is with the data when you need it. To be fair, many of the leading data integration tools were first created before the advent of XML (the draft specification for which was published in 1996—it was accepted as W3C recommendation in 1998) and the use of a metadata repository seemed like the only logical approach. More recent products to enter the market have simply not questioned the status quo and, as far as we know, have not considered the potential of the implicit metadata in XML. With hindsight this appears to have been a mistake.

Of course, a lot of data is not in XML format. However, it is not hard to convert relational data, say, into XML format. It is not even a transformation, it is a conversion and, unlike transformations, conversions are easy to automate, either at the connector level or within the database itself. So, let us consider what an ETL process would look like in XML terms. There would be the following steps:

1. Where necessary, convert from relational or other source format into XML.

2. Where necessary, convert from relational or other target format into XML.

3. Map from one XML format to the other.

Of course, the secret sauce in this is being able to map from one XML document to another. Historically it has not been thought possible to do this in any sort of automated fashion. However Tralee Software has patented methods to achieve this and has a product, ETLmessenger (see next section), which demonstrates that this can actually be achieved. This means that you can:

- Automate the extraction process
- Automate the load process
- At least partly automate the transformation process
- Perform the transformation process in memory without having to access a metadata repository on disk, thereby improving performance and supporting scalability
- Reduce the size of the software stack (you don't need separate ETL, EAI, CDC components)
- Reduce complexity and administration requirements
- Eliminate (often) the need for a developer or IT involvement, thereby reducing development time and reducing costs
- Future-proof against forthcoming developments such as the semantic web
- Provide support for industry meta-models such as XBRL.

While Tralee has utilised its technology to address many of the core issues facing data integration, the company believes the technology has broad utility across other applications and is embarking on at least half a dozen new research and development projects to extend that utility to next generation user requirements.

ETLmessenger

ETLmessenger is a prototype data integration platform, although it has been installed in live sites and proved its capabilities. The way that ETLmessenger works, with respect to the points made in the previous section (plus additional details), are as follows:

1. Extract automation: if the data is in XML format then it is extracted directly. For relational data and other environments supporting SQL access, data is extracted via a SQL query running over ODBC/JDBC. Reusable SQL templates that generate these SQL queries are provided out-of-the-box. These templates are stored in an XML formatted node map.

2. Load automation: the reverse of extraction.

3. Transformation automation: this is achieved via XML mapping files that map from the node map of the source to the node map of the target. Existing XML DTDs can be reused where appropriate. Note that the entirety of each record is loaded into memory and transformed there. There are no field-by-field I/Os and there is no need to access metadata.

4. A graphical user interface (Eclipse-based) is provided so that users can customise SQL templates and XML mapping files. Flexible mapping capabilities are supported, such as field splits and concatenation, record-at-a-time processing, set-level processing and so on.

5. The product uses an internal in-memory data store for mapping operations. Source data is written into this store, transformed and then written to the target.

6. In certain environments it may not be practical to do all processing within ETLmessenger: for example, if you want to run data cleansing processes as a part of your data integration. In this case, ETLmessenger can be used as either a pre-processor or a post-processor to the data integration engine, or both. The same might apply where, for performance reasons, you might wish to push processing into a data warehouse, or where you are dealing with data that is neither XML-based nor susceptible to SQL queries (for example, SWIFT messages).

7. XML data is transmitted between source, intermediate and target systems using JMS, HTTPS, SMTP or FTP protocols and the product supports distributed topologies. A lightweight messaging capability (MQLite) is included within the product but other JMS compliant messaging systems may be used.

8. ETLmessenger is agnostic as far as latency is concerned. It can be used in a conventional batch environment, with micro-batches or in conjunction with change data capture (which works by polling the source), as required.

As mentioned, ETLmessenger is a prototype, designed to illustrate the fact that Tralee's patents are genuine and really works, as much as anything else. As such, Tralee has not developed the product more than it has needed to demonstrate its capabilities. Thus there are a number of areas in which ETLmessenger might be usefully extended. In particular, support for XQuery, XPath and XSLT could all be incorporated in order to provide extended transformation capability and flexibility. Integration with IBM's pureXML technology would also be beneficial for DB2 users. Additional enhancements might include integration with products such as Altova's XMLSpy for XML validation and adoption of the OASIS CAM (content assembly mechanism) standard. The latter is designed to support B2B environments using XML and also enables look-ups against external files, which would allow such things as postcode validation.

Because ETLmessenger runs entirely in memory, another potential enhancement could be to implement it in an in-memory database such as IBM solidDB or Oracle TimesTen. Further, something that can be automated can be embedded into silicon. It is therefore entirely conceivable that one might develop an 'integration chip' that would further enhance performance. We understand that Tralee is in discussions with Intel and Rambus on this subject.

The competitive landscape

All of the vendors currently in the data integration market have repository-based products. ETLmessenger is the only technology (at least that we are aware of) that uses an XML-based approach and which eschews the use of a repository. The principle advantage of the former approach is that the deployment of a repository is useful for multiple reasons—not just data integration. For example, you would want to use a repository to support a business glossary, to support impact and where-used analyses, to determine dependencies, to store relevant models and so on.

So, we are not suggesting that you can dispense with the use of metadata or repositories. What we are suggesting is that the use of such a repository is unnecessary in order to support the movement and transformation of data. Using an XML-based approach, as outlined in this document, will be much simpler, easier to use, and should scale and perform better.

One question that might be asked is whether XML will be suitable in all cases. How suitable is the existing file-to-database and database-to-file conversions for non-standard environments such as handling spreadsheet data or loading data into a NonSQL database rather than a relational one. In practice, we see no reason why the technology should not be extended to cater for such inputs and outputs. Indeed, Tralee has demonstrated that it is possible to use its software in conjunction with interchange formats such as SWIFT and EDI messages.

The one place where Tralee's approach may not be suitable is when very complex transformations are required. However, companies that have such requirements are most likely to have data integration suites in place already and are less likely to be companies employing hand coding.

The main reasons why users still hand code data integration solutions is that they think it is easier and cheaper to do that. If data integration vendors continue on their current path there is little likelihood that they will ever persuade these users otherwise. The XML-based solution developed by Tralee Software potentially resolves that conundrum, by providing functionality with ease of use at a lower cost. Note that we don't mean lower cost with respect to existing integration platforms but with respect to hand coding. XML offers the potential to automate data integration processes, thereby requiring no coding and no time spent in maintaining that (non-existent) code, and that's cheaper than doing it manually.

Conclusion

We do not contend that the methods used within ETLmessenger represent a panacea. However, it does address some of the major issues faced by conventional approaches to data integration. In particular, there is no need for a metadata repository to support ETL/ELT processes (though you may still require such a repository for other purposes; to support a business glossary, for example). As a result there are far fewer overheads in terms of both administration and run-time performance. In addition, the fact that metadata is implicit means that this approach is essentially declarative rather than procedural: it takes data integration a long way down the path towards automation and the removal of complexity. The bottom line is that this approach requires far less in the way of resources, particularly with respect to personnel but also hardware and software resources, and therefore costs, than traditional methods of supporting data integration. In particular, whereas most data integration vendors have separate products (albeit part of a platform) in order to support ETL, B2B, EAI and so forth, Tralee offers a solution that caters for all of these within a single product. As a result of all these considerations, ETLmessenger and its planned cloud-based derivative web integration service are likely to prove attractive to all classes of user and, especially, to those in the mid-market that have previously eschewed the use of data integration tools.

We are advised that Tralee is in discussions with various data integration vendors and other potential partners that have expressed an interest in licensing its technology and assisting the company to bring its technology into the mainstream. We can only recommend such an action: the Tralee technology will not only simplify existing environments for new and existing users but it will also mean that licensees are well-positioned as we move into the era of the semantic web, which will be dominated by RDF/XML. Needless to say, we conclude that any vendor that does not take up such a license, or think of some other way of achieving the same results (which might be difficult given Tralee's patents), is going to be at a disadvantage compared to those that do.

Further Information

Further information is available from
http://www.BloorResearch.com/update/2137

Bloor Research overview

Bloor Research is one of Europe's leading IT research, analysis and consultancy organisations. We explain how to bring greater Agility to corporate IT systems through the effective governance, management and leverage of Information. We have built a reputation for 'telling the right story' with independent, intelligent, well-articulated communications content and publications on all aspects of the ICT industry. We believe the objective of telling the right story is to:

- Describe the technology in context to its business value and the other systems and processes it interacts with.
- Understand how new and innovative technologies fit in with existing ICT investments.
- Look at the whole market and explain all the solutions available and how they can be more effectively evaluated.
- Filter "noise" and make it easier to find the additional information or news that supports both investment and implementation.
- Ensure all our content is available through the most appropriate channel.

Founded in 1989, we have spent over two decades distributing research and analysis to IT user and vendor organisations throughout the world via online subscriptions, tailored research services, events and consultancy projects. We are committed to turning our knowledge into business value for you.

About the author

Philip Howard
Research Director - Data Management

Philip started in the computer industry way back in 1973 and has variously worked as a systems analyst, programmer and salesperson, as well as in marketing and product management, for a variety of companies including GEC Marconi, GPT, Philips Data Systems, Raytheon and NCR.

After a quarter of a century of not being his own boss Philip set up his own company in 1992 and his first client was Bloor Research (then ButlerBloor), with Philip working for the company as an associate analyst. His relationship with Bloor Research has continued since that time and he is now Research Director focused on Data Management.

Data management refers to the management, movement, governance and storage of data and involves diverse technologies that include (but are not limited to) databases and data warehousing, data integration (including ETL, data migration and data federation), data quality, master data management, metadata management and log and event management. Philip also tracks spreadsheet management and complex event processing.

In addition to the numerous reports Philip has written on behalf of Bloor Research, Philip also contributes regularly to IT-Director.com and IT-Analysis.com and was previously editor of both "Application Development News" and "Operating System News" on behalf of Cambridge Market Intelligence (CMI). He has also contributed to various magazines and written a number of reports published by companies such as CMI and The Financial Times. Philip speaks regularly at conferences and other events throughout Europe and North America.

Away from work, Philip's primary leisure activities are canal boats, skiing, playing Bridge (at which he is a Life Master), dining out and walking Benji the dog.

Copyright & disclaimer

2nd Floor,
145–157 St John Street
LONDON,
EC1V 4PY, United Kingdom

Tel: +44 (0)207 043 9750
Fax: +44 (0)207 043 9748
Web: www.BloorResearch.com
email: info@BloorResearch.com