FEEDING THE ENTERPRISE DATA REPOSITORIES

Bogdan Pilawski
Bank Zachodni WBK, Strzegomska 8-10, 53-611 Wroclaw, Poland
e-mail: bogdan.pilawski@bzwbk.pl

ABSTRACT:

The enterprise data repositories are becoming more and more important source of information, on which the strategic and management decisions are made. That rises also the importance of how they are fed with data, and what is fed into them. This paper is based on current, however initial, practical experience, and discusses and compares selected methods used to feed data into data repositories. It also raises some data quality related issues, especially in relation to organisations dealing with mass customers, like financial institutions.

This paper undertakes to introduce readers into the subject of ETL, which, despite numerous implementations, remains still relatively unknown, even among members of IT community. For that same reason this paper does not delve deeper into the matter, what might be disappointing for those, who are already familiar with basics of the subject.

KEYWORDS

Data, data repository, ETL Tools, ETL process, data warehouse, data quality

INTRODUCTION

The companies and businesses engaging themselves into implementation and running of the wide scale data warehousing (business intelligence) solutions are facing numerous obstacles which hamper their effort and slow down their progress towards achieving meaningful business effects from that technology.

One of the issues is the efficient feeding of the data warehouse with up to date data, from multiple sources of different structures and of various internal organisations, update cycles and sizes. More to that – there are no easily available, ready-made software means to control and to ensure that all data records to be moved into the data warehouse, are getting there on time or even at all.

The impairing data structures of various data sources involved are another source of concern, when put against a data warehouse data model. Numerous enterprises were taught a painful lesson while trying to map the source data structures onto those of the data warehouse, in a shortcut attempt to reach the theoreticians praised “end-to-end metadata” kind of solution.

Besides the data warehouse internals, there are two major areas likely to result in continuous production problems, if not put right from the outset. They seem to embrace the data warehouse area somehow (in a wide sense), since one of them concerns feeding the data warehouse from source systems, while the other relates to the extraction of various sets of

data from the warehouse, to enable for analytical and reporting functions to be carried on them.

According to sources quoted in the [Adastra_2005] leaflet, feeding the data warehouse from its sources became the most critical and costly part of any information system designed for business intelligence. The same source recalls the opinion of Gartner ETL specialists, saying that people working in this area alone comprise 45% of an organisation's data warehousing staff. Gartner experts also insist that more than 50% of all development work for any data warehousing project is spent on ETL, while in the area of design and development of data mining applications, more than 80% of all work is spent on data preparation. Those numbers seem to constitute a good argument, enough to justify solutions where the tool-based ETL solutions are applied, instead of home grown software.

ETL SOFTWARE PACKAGES

In the early days of data warehousing the process of feeding the data warehouse with data from their sources was commonly served by the in-house written set of software modules. Various software creation techniques were applied, and usually any change in any of the data sources involved, required programmers intervention and subsequent re-testing. That was not only ineffective in many aspects, but also error-prone and long lasting. The data flow resulting with this approach, from source repositories up to the final business intelligence solutions involved, is presented on picture 1.

According to Gartner's paper by T. Friedman [Friedman_2001] "*data acquisition is often the hardest part of a data warehouse effort due to:*

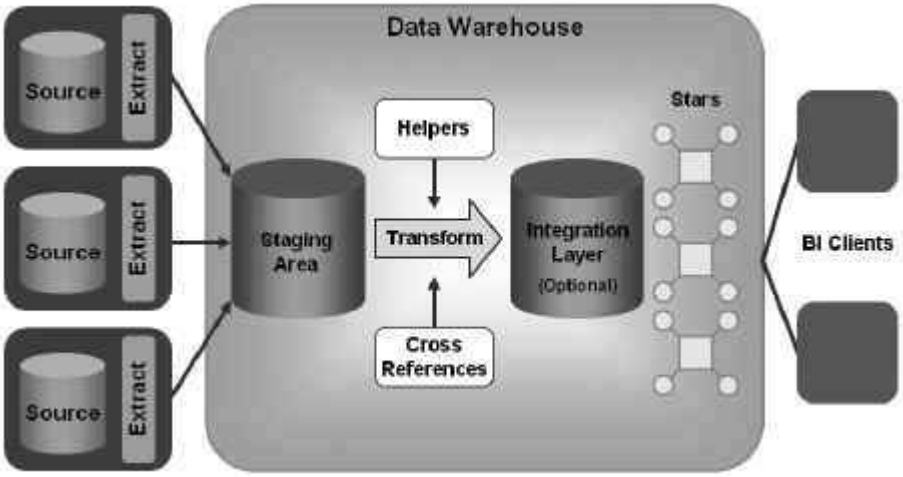
- *The large number of disparate data sources that must be tapped to pull together the data of interest to the enterprise*
- *Lack of available resources with detailed knowledge and documentation of legacy system functionality and data structures*
- *Data quality issues within operational systems, as well as unintended uses of data elements due to application changes*
- *Unrealistic time frames for data analysis and development of extract/transformation/loading (ETL) processes".*

Friedman insists, the typical large enterprise nowadays runs an average of eight different database management systems, and its data are spread over an average of fifty operational systems. The task of bringing the data from those disparate sources into one unified repository requires specialised skills and technology to deal with the different syntax of each database system. Differing semantics bring further complexity into that.

The problem of the said complexity is also being raised by others. The [Informatica_2006] paper says, that *"Data transformation functions are usually fragmented across many technologies, and may embrace data transformation logic in many different places, e.g., in brokers, adapters, extract transform-load (ETL) tools, and most importantly, in expensive and error-prone custom integration solutions code. This means that even successful data transformation efforts may not be reusable or maintainable."*

The accumulated difficulties faced by organisations while feeding the data warehouses with data, have triggered the actions to design the specialised software tools, aimed to ease the situation, and to tackle the problems concerned head on. Its obvious the task has been tackled applying all the knowledge and experience gathered so far, while struggling to keep home-made solutions up and running.

Picture 1. Data flow in the non-ETL business intelligence solution



Source: [Adastra_2005]

Throughout the years numerous specialised software tools became available, capable not only to transfer the source data into the warehouse in an organised way, but also to facilitate the creation and maintenance of software involved, by means of user-friendly graphical interfaces.[1] The software used for the purpose is now known as Extract-Transform-Load (or ETL for short) to indicate three main function it is capable of performing. One must stress, the software of that kind, while capable of relieving many data transportation and transformation pains, will never do its task without some highly specialised, preparatory work. This work however is limited to higher level activities only, is less error-prone and takes significantly less time.

The latter is also clearly expressed by Gartner Research's expert, Ted Friedman. In the [Friedman_2001] paper he insists, that "*Enterprises have a natural tendency to assume that technology alone will solve these problems. Although ETL tools can help, they do nothing to address the issues of identifying where the source data resides, assessing its level of quality or defining the business rules for how it must be cleansed and integrated before loading it into the data warehouse*".

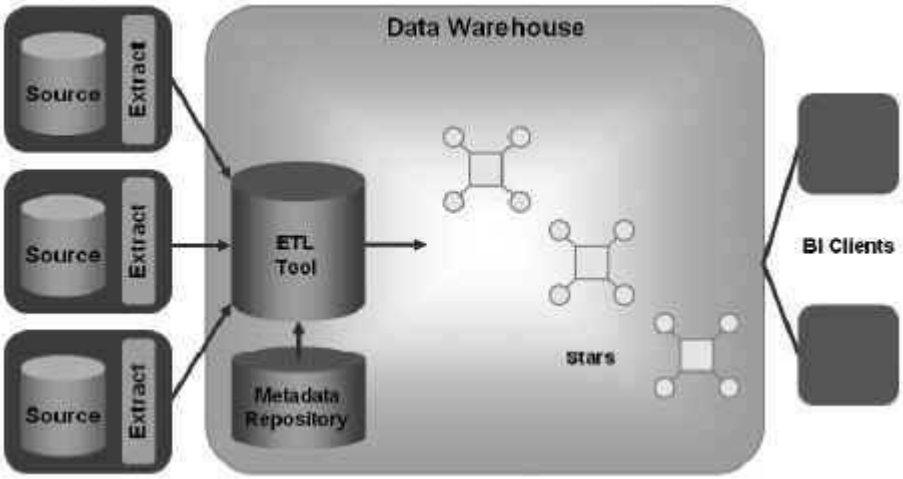
Picture 2 is a good example how specialised ETL tools compare to the manual solutions of that kind presented on picture 1, however the differences visible are only part of what is really available with ETL tools.

[1] "*Most ETL tools use a graphical programming paradigm to shield the user from the complexity of coding the transformations and make the tools easier to learn and use*" [Sunopsis_2006]

The functions of the software discussed here are in fact not limited to extracting data from their sources, and putting them into the target repository, be it e.g. data warehouse. On their way between the two one performs also:

- Various checks and controls to prevent the erroneous data to enter the repository,
- Controls to ensure the data sets transported are consistent within themselves (i.e. there are no contradictions between various elements of data),
- Final checks to ensure the data about to enter the repository are "clean", from both – business and IT points of view.

Picture 2. Data flow in the ETL-based business intelligence solution



Source: [Adastra_2005]

Although all above requirements are equally important, it is that calling for only "clean" data to be allowed to enter the repository, which seems to be of the utmost importance. The very concept of data cleanness is not precisely defined, and various sources present different views on that matter.

While one of these views wants the data entering the repository to conform to the rules resulting from the attributes and characteristics of each individual data field defined in the repository's data model, the other go even further and require for the content of entry data stream to be consistent with rules and patterns defined and maintained outside the scope of the data repository and its model of data.

To good example for that is the postal area delivery number, or zip code, as it is commonly known. One kind of requirement might be for that number to consist of digits only, allowing in fact any combination of digits to be entered as the zip code and accepted. The other approach would require to have this code also vetted against the database of all zip codes currently present in a given geography, and also against other parts of address, like name of the town, street name (and also home number in this street) within that town. Checks of similar nature can be made against predefined lists of names of the locations (towns, cities), or country specific sets of customer christian names, family names etc.

Checks of that kind however, if performed, can be regarded only as a last resort means of prevention, as not to allow the inconsistent data into the repository. The proper data cleansing

process cannot be done fully automatically, so it is usually regarded as a separate activity, performed on operational data repositories, where the human support to resolve issues encountered can be provided. E.g. the address versus postal zip code inconsistency can not be resolved automatically by assumption that the address field content is always correct, and the zip code should be adjusted accordingly (or vice versa), if it turns out to be inconsistent. The issue like this can only be made clear by verification of those both components against each other, and that usually can not be done without human intervention.

For that reason the ETL process will be discussed here as being performed on already clean data, which do not require any further attention in this respect.[2]

ETL SOFTWARE IN OPERATION

The sources of operational data are various in nature, and can be anything from flat files to databases of different origins. Those sources also do vary in size of data to be transferred during a single transfer-session.[3]

The other difference can be the way the data are qualified for transfer, and especially whether the source of data has any means available to mark and hence involve in transfer only the data added or updated since previous session of transfer. The facility to indicate data of that kind can significantly reduce thee duration of the session, and also lessen the load burden put on both – the source data repository, and also on the target repository, whether this is data warehouse proper or any other kind of intermediate, temporary data storage.[4]

As it has been put in short but important paper by Massimo Cenci Cenci_2002], the extraction of data from their sources in practice is usually limited to the 1:1 replication to staging area tables.[5] That approach ensures the extraction process becomes relatively short in duration and puts only the minimum load on source system. Any further processing and transportation of data does not involve the source system in any way.

The main advantage of a purchased solution over its home made counterparty is, that the former is capable of exploiting the so called native interfaces when accessing both, the sources of data, and the target data warehouse database. The native interfaces manage to avoid the additional load usually introduced by SQL or ODBC kind of data manipulation tools, by getting deep into intricacies and peculiarities of physical data access, which is characteristic and specific to a given kind of source data. For that reason every single native interface is manufacturer-specific, and will not work at all with any other data source, except for the one it has been designed for.[6]

So bringing in the specialised ETL software tools not only eliminates the need for manual coding (and a consequential testing of what has been coded), but also introduces the well

[2] the subject of data quality is discussed in more detail in the [Laney_2002] paper
[3] This is meant as a repeatable process, performed in pre-determined time intervals (e.g. hourly, daily, weekly etc.), and, when initiated, running uninterrupted until its planned action is completed
[4] in the branch jargon this is often called “staging data area”
[5] that approach becomes more and more often questioned, and there are also voices calling to give up the whole idea of staging the data; for the sake of higher efficiency at lower cost; this new approach is discussed in more detail at the end of this paper
[6] this is somehow similar to the concept of so called printer drivers, which are to be found in almost every computer, and which are a kind of go-between for various activities

proven and tested data access means, capable of providing much higher overall data processing efficiency.

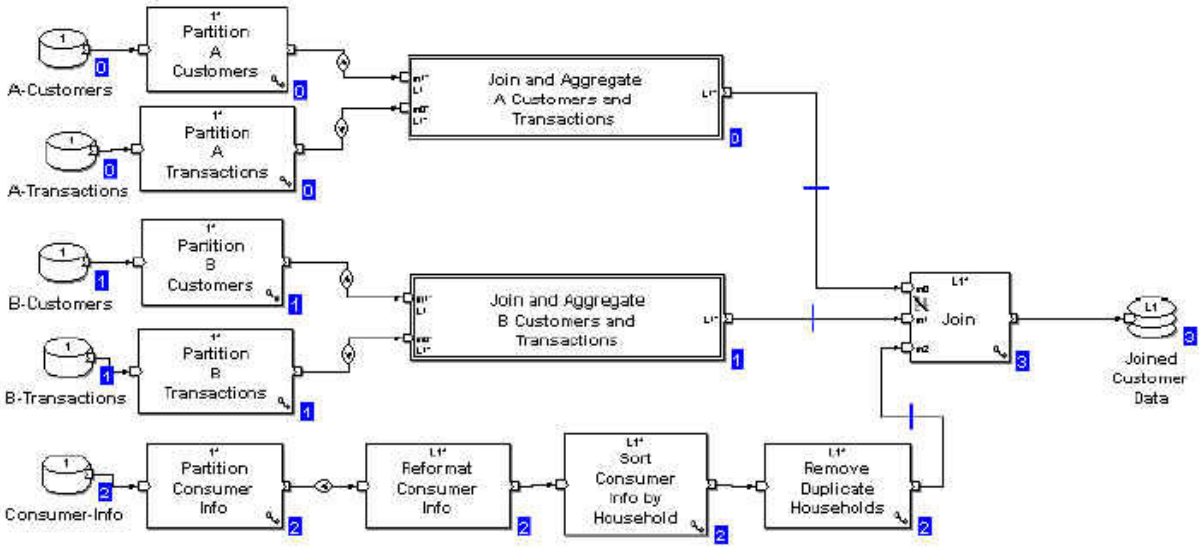
The purchased software usually not only provides better efficiency in production, but also allows for easier and carried in a controlled manner introduction of changes.

The extraction of data from their source systems carried with ETL tools usually is just the beginning of the process aiming to bring those data into the data warehouse. However there is noting in the way what would have prevented the usage of those tools outside the data warehouse context, e.g. to accomplish a pure transformation of data of a given structure into the set of data of another structure.

Putting aside the call for the highest possible efficiency of the extraction process itself, there is nothing against performing any transformation of data required, while transporting them between their source and staging repositories.

The way the data are transformed using the ETL tools depends on the needs and on the conditions the data in question are capable of withstanding. The simplest form possible would be the pure transfer of data from its source to destination container, with some re-mapping of data fields during the process. More complex task would involve combining multiple sources into a single, merged output stream. The opposite to that would be the case, when the data from a single source are output to a number of target containers according to some criteria evaluated by the ETL tool. The sample developer screen presenting the ETL tool-generated ETL process diagram is presented on picture 3.[7]

Picture 3. Screen-presented diagram of the typical ETL process



Source: BZWBK Bank test data

Finally, the most complex transformations would involve various content checks and controls, re-calculations and trans-coding. Accordingly as a result of this operation the data items would either be placed in a destination container selected based on the content of the data in

[7] screen shots presented on pictures 3, 4 and 5 in this paper originated from the Co>Operating System ETL tool of Ab Initio Software Corporation

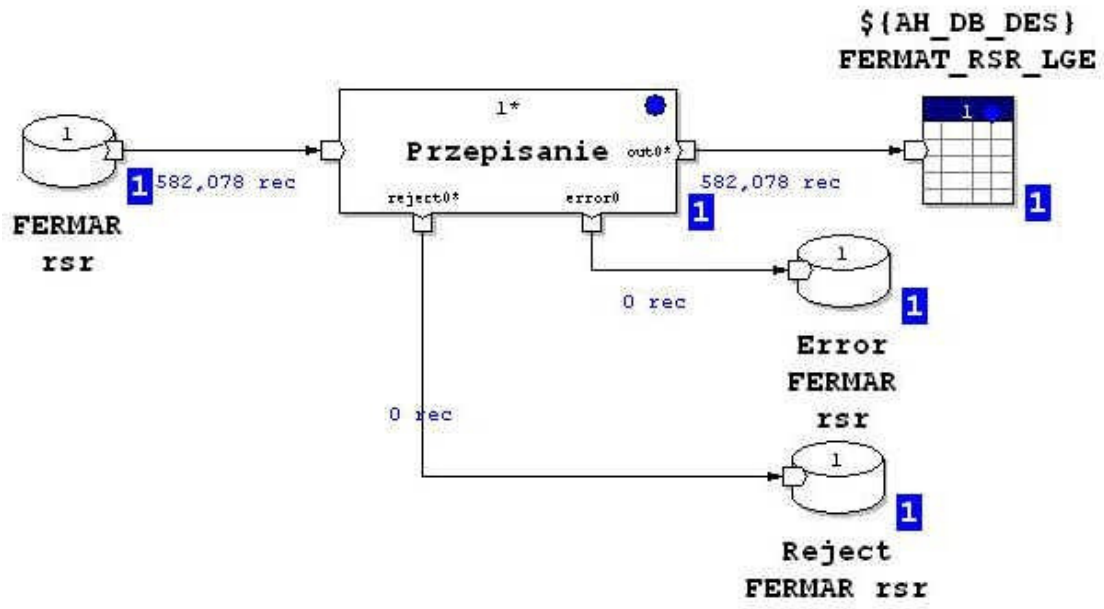
question, or would enter the data in error containers, which in their simplest form might take the form of error logs or journals.

The ETL tools are capable of maintaining the record counters associated with every single data container, whether input or output, enabling for operational controls to be applied and carried out.

The transformations are designed and tested using the graphical interface provided along with the tool, so there is almost no need for any manual coding to be done. The function to perform various checks, controls and the data transformations proper can be easily set up and tested while still in the design stage, using ad hoc or real data. This important software development facility embedded in the ETL tools simplifies and speeds up the design process, and allows for early discovery of errors and their subsequent clearing. However this testing of particular functions can not be seen as a substitution for overall testing of every solution designed, taken as a whole.

The sample (and relatively simple) transformation is presented on picture 4. The flow of data is from the left to the right, and the named boxes represent the data containers involved. The numbers associated with them stand for quantities of data records processed in a particular run. The records qualified as not fulfilling output container (here: FERMAT_RSR_LGE) entry conditions, are diverted to error log (here: Error FERMAR rsr) and to the rejected records container (here: Reject FERMAR rsr).

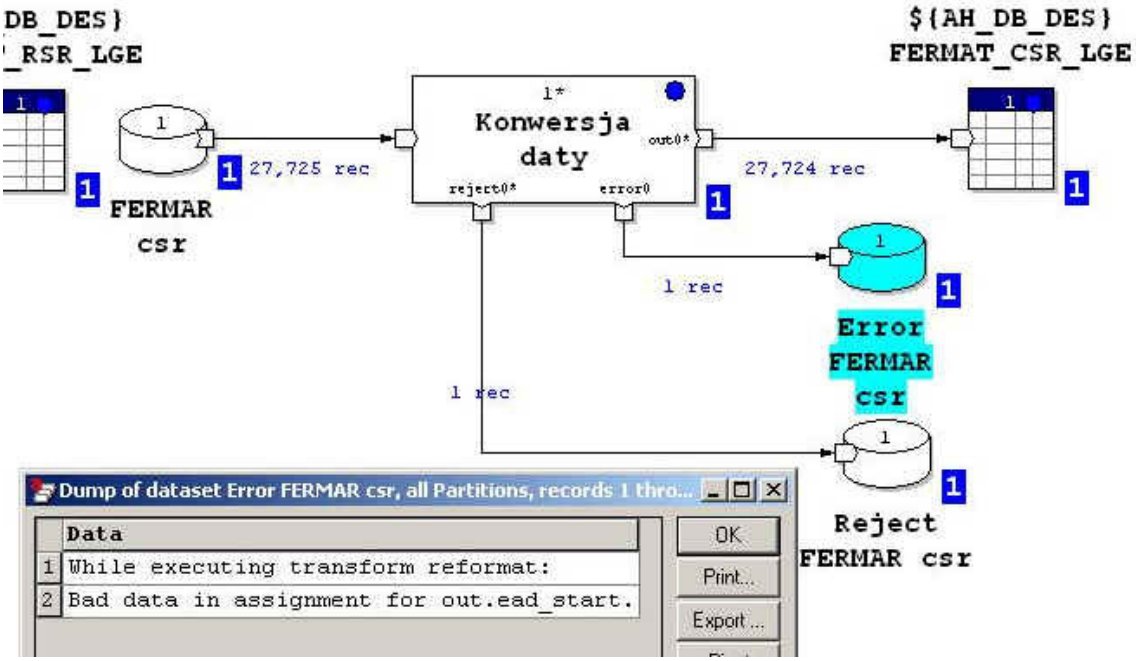
Picture 4. Screen representation of sample ETL production run



Source: BZWBK Bank test data

Picture 5 illustrates another transformation run with part of error handling screen shown..

Picture 5. Screen shot of software developers work using ETL tools



CLOSE

The practical implementations of ETL tools[8] have now reached a stage of some maturity, and in most cases it seems there no doubts about how efficient can they be (see picture 6 which shows comparison of time taken to run non-ETL and ETL tests on identical sets of data). For some time now however the new concept of how data warehouses can be fed with data are raised. Within those the very idea of ETL is being criticised, as introducing the intermediate stage, now regarded obsolete. That stage is said to claim and engage unnecessary resources, increasing the risk of the process as a whole, and exploiting the additional specialised skills.

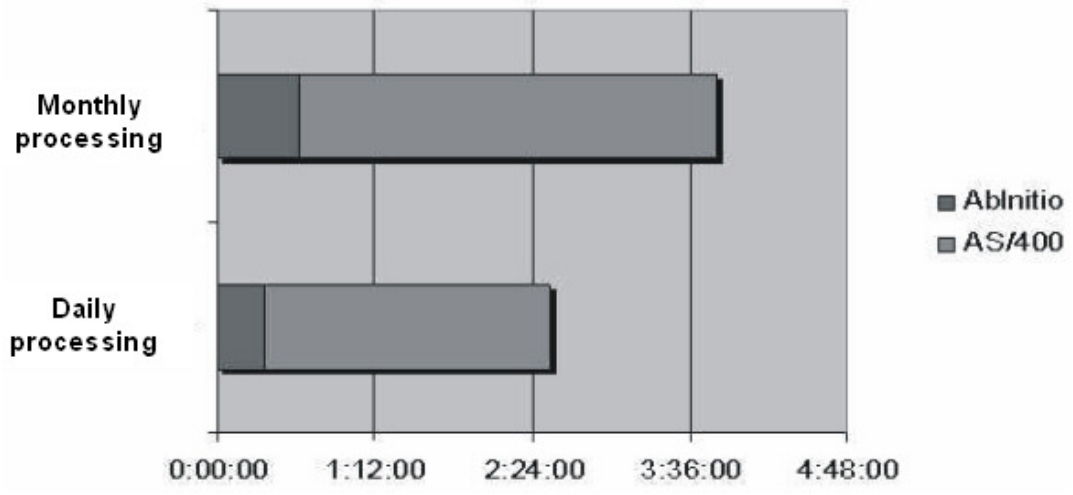
All this leads to the new concept which still relies on three separate actions performed on data on their way from source systems up to the data warehouse proper. The difference is that the data is loaded into the warehouse first, and transformed to the shape required afterwards.

This whole E-LT concept, as it is called, is presented in detail in the [Sunopsis_2006] paper.[9] While it undertakes to defend the new approach as less resource hungry and more cost effective, it does not take into account some important facilities available with traditional ETL tools nowadays (like e.g. already mentioned in this paper multiple native interfaces, capable of performing input-output operations on both – source data and target data warehouses in a way which is most efficient for a given combination of source and target database internals.

[8] ETL products available from various software producers are discussed and compared in the [Eckerson_2003] paper

[9] it is also discussed in [Linstedt_2004]

Picture 6. Results of non-ETL and ETL test runs performed on the same sets of data



Source: BZWBK Bank test data

It is also data warehouse software manufacturers, who adapt their products to be better ETL-suitable. E.g. NCR's Teradata Database compression is completely internal to the system, in a way which makes it transparent to ETL operations, queries using base table access, queries using view access, and all application software. [Teradata_2003]

It seems both ETL and E-LT data feeding approaches will co-exist in parallel for some time. What can bring the real change into the area (and along with it a number of entirely new challenges to cope) is the concept of the active data warehouse, which is already known and around for some time.[10] Switching to this kind of solution will enforce usage of entirely new methods and operational concepts, while giving up the very basic idea of periodic data updates. For this to take place however, those new solutions need to prove they can bring the business effects well capable to justify spending they will require.

LITERATURA

[Adastra_2005] *Business Intelligence Solutions - Tool Based ETL*, Adastra Corporation, Ontario, 2005

[Brobst_2003] Brobst, Steven, Ballinger, Carrie, *Active Data Warehousing*, Teradata Inc., 2003

[Cenci_2002] Cenci, Massimo, *The Infrastructural Datawarehouse*, The Data Warehousing Institute, 2002

[Eckerson_2003] Eckerson, Wayne, White, Colin, *Evaluating ETL and Data Integration Platforms*, The Data Warehousing Institute, 2003

[Friedman_2001] Friedman, Ted, Strange, Kevin H., *Staffing Data Acquisition efforts for the Data Warehous*e, Gartner Research, 2001

[10] the concept of active data warehouse is presented in more detail in [Brobst_2003] and [Graham_2006] papers

[Graham_2006] Graham, Dan, *Enabling the Agile Enterprise with Active Data Warehousing*, Teradata Inc., 2006

[Informatica_2006] *The Complexity Advantage: Solving the Requirements of Complex Data Transformation in Business Integration*, Informatica - The Data Integration Company, Redwood City, 2006

[Laney_2002] Laney, Doug, *Ascending the Information Maturity Model: Part 1. Data Quality*, META Group, 2002

[Linstedt_2004] Linstedt, Dan, *Additional thoughts: ETL and ELT, RDBMS (Part1 and Part 2)*, Teradata Magazine, September 2004

[Sunopsis_2006] *Is ETL Becoming Obsolete?,* Sunopsis Inc., 2006

[Teradata_2003] *Teradata Database - Database Design*, Teradata Inc., 2003