Important Considerations for an ETL Process in CFE's Health and Security Area

J. Roblero, N. Jácome and L. Argotte

***Abstract*— Today information plays a key role as competitive advantage for organizations. "Comisión Federal de Electricidad" (CFE the electricity utility in Mexico) is not an exception, and as part of its modernization process they implement tools in order to improve their information about the security's process management in a more organized way with high quality standards. In this paper, you will find important considerations for Business Intelligence (BI) to enable those people in charge of security the access to information at the right time in order to take better decisions based on data. Specifically, it presents the process of Extraction, Transformation and Load (ETL) of security's data, focusing on methodology required to ensure quality information.**

***Index Terms*— Business Intelligence, Extract, Load, Transformation**

I. Introduction

BEST practices in BI propose to follow a development methodology. Specifically for ETL's packages there are different mechanisms in order to work with an ETL's process; Kimball's classical approach [1], adaptable transformation [2], hybrid approaches, and others.

According to Vassiladis [3], extraction, transformation and cleaning problems may consume 80% of time employed in data warehouse development. Many organizations made a mistake by implementing a solution without taking in consideration the importance of a following model. This work is based in methodology proposed by Inmon [4] to create a solution that supports the decision's making process in industrial security area.

Since 2007, CFE counts with an Integral Security System and work's health (SISST in México) that manages the hygiene and security's process at work. The reports designed in the system offer to the users operational information about the whole process. When this process was automatized executive summaries with consolidated data weren't needed, however in 2009, it was outlined the need of having indicators and information from data generated during this process. This is why is necessary to integrate and analyze them.

Manuscript received June, 2011; revised July 23, 2011. This work was supported by the "Instituto de Investigaciones Electricas" of Mexico.

J. Roblero is with the Instituto de Investigaciones Electricas 62490 Cuernavaca, Morelos, Mexico (e-mail: jroblero@iie.org.mx).

N. Jácome is with the Instituto de Investigaciones Electricas 62490 Cuernavaca, Morelos, Mexico (e-mail: nejacome@iie.org.mx).

L. Argotte is with the Instituto de Investigaciones Electricas 62490 Cuernavaca, Morelos, Mexico (e-mail: largotte@iie.org.mx).

II. ETL'S PROCESS IN SECURITY AREA

A. *Previous attempts*

1) The first approach with ETL's tool consisted in extracting information from SISST's database, having as a major goal to generate an analysis with an Excel's Pivot Table. To be able to do this all the accident's tables were imported from the system and each time the information was loaded, the necessary tables were removed and rebuild from data warehouse. This is a classical example of malpractices done as a consequence of and ETL projects.
2) The most common idea designing a business intelligence project is to concentrate information without worrying about performance, and afterwards mine it in any possible way. This creates an overloading in productive system collapsing during the process of uploading information. This solution provided a view of information that even though it is useful for the user it did not worry about best practices for a BI solution.
3) An improvement to the previous solution was to use a single package or integration component that was responsible for the entire ETL process.
4) While designing this package was made several mistakes, the most important were:
   - Whole data was imported from productive tables and calculations were performed directly on them.
   - A design with low modularity complicates the maintenance, because a single package manages the extraction, transformation and loading of all data that field the data warehouse.
   - The data warehouse was erased and then a complete import of data was done in each load.
   - The cube was processed on the same package.
   - *Merge join* component was used to upload information, instead of *Look up* component in order to use fewer resources.
   - *Merge join* component requires a previous data sorting, *Sort* component consumes a lot of resources, and its use should be avoided as much as possible, as seen in the Fig. 1.

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

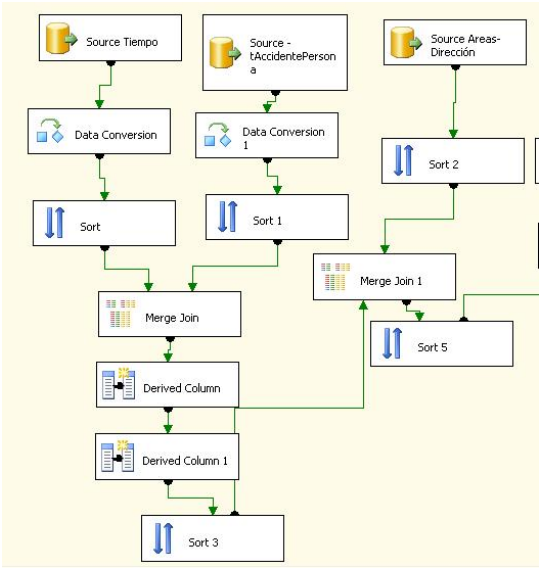

Fig. 1 *Sort's* component usage

All these mistakes made the maintaining highly inefficient and the solution's performance even more inefficient as well.

III. INTEGRATING THE SOLUTION

The ETL ´s process steps and the specific characteristics to be taken in consideration to minimize the solution's implementation time, will be detailed in this section.

The solution structure to upload the information into the data warehouse is configured in the following way:

As a first step global data sources for packages were defined, this facilitates the solution deploy process, because these sources are shared with each package and setting the deployment utility will be done only once instead of setting them for each package.

For the data warehouse filling process it was necessary to divide the solution in several packages, easy to understand and maintain if necessary. Each dimension from the data warehouse and the fact table, were traduced into a single package within the ETL as seen in Fig. 2.

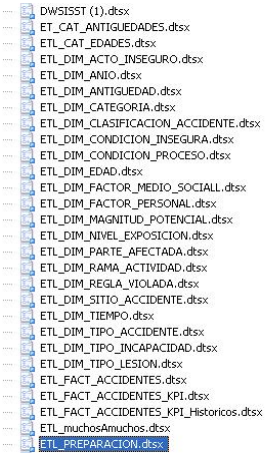

Fig. 2 ETL's structure.

Besides this, five additional packages were created to encapsulate and determine execution order for the rest of the packages as well as the needed variables for each one, to make, in an organized way, the information's loading of its source to its destination.

Data profiling

This activity must be the first one in an ETL'S project, having as main goal to know the structure from the source's information. The most common tasks offered by several softwares in profiling area are:

- Candidate Key: A profile that reports whether a column or set of columns are appropriate to serve as a key for the selected table; it also verifies the existence of another candidate's key and if there are any violations.
- Column Length Distribution: It gives a minimum and maximum column length view, more detailed for each column, it shows the rows percentage that for filed a determinate pattern.
- Column Null Ratio: It shows each table columns and the null ratio percentage, this function is very useful when you desire to transform this type of values into something more significant.
- Column Pattern Profile: It shows the pattern or the set of patterns that contain the column's data, based on regular expressions.
- Column Statistics Profile: Applicable only for numeric data (integer, float, decimal, etc.) and dates (dates only allow minimum and maximum calculations). Numeric calculations are: Minimum, maximum, average and standard deviation.
- Column Value Distribution: Shows each table column, the number of different values, each column distribution value, as well as the percentage or rows that contain it.
- Functional Dependency: A Functional Dependency profile reports the extent to which the values in one column (the dependent column) depend on the values in another column or set of columns (the determinant column). This profile can also help you identify problems in your data such as invalid values. For example, the dependency between the Postal Code column and the state column. In this profile, the same Zip Code should always have the same state, but the profile discovers violations of the dependency.

Until this point the possible types of analysis and their utility has been outlined. The data profiling task allows us to identify the quality and range of values in the source. This is extremely important for a data integration project, knowing in detail the structure and the kind of information contained in data sources, improves ETL's package design.

IV. FACT TABLE AND DIMENSIONS DATA STORAGE

A. *Dimensions*

Dimensions give the context to the fact tables, and therefore every measure in the data warehouse. Dimension table attributes play a vital role in the data warehouse. Since they are the source of virtually all interesting constraints and report labels, they are the key to making the data warehouse usable and understandable. In many ways, the data

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

warehouse is only as well as the dimension attributes [5].

The "Dimension's" term is used in some BI applications as equivalent to "hierarchy" (especially in multidimensional databases). So, a geographical dimension includes different levels, like continents, countries, regions, provinces and localities.

A. *Slowly changing dimensions*

According to Kimball [5], there are three kinds of slowly changing dimensions:

SCD Type 1 Overwrite the Value: When an attribute's value change, the old value is overwritten without saving history. This means that whole history is missing, and if you want to make an analysis, information will be presented with the actual value.

SCD Type 2 Add a Dimension Row: When an attribute value changes, a new record is added to the table to represent the new information. Therefore, both the original and the new record will be present. The newest record gets its own primary key. A "version" or "date" field is used to indicate the last record in the dimension (Fig. 3).

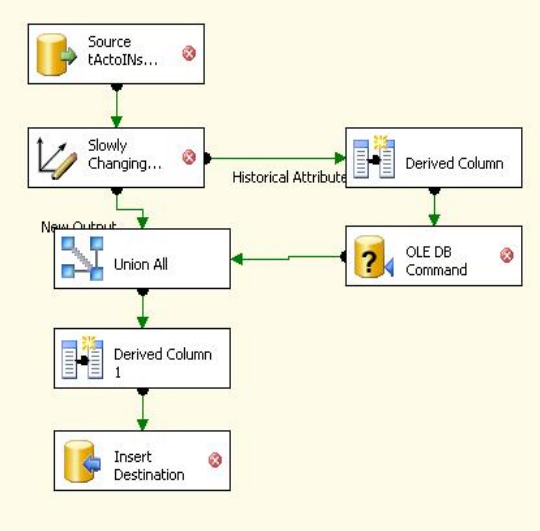

Fig. 3 SCD type 2, dimension history is maintained

SCD Type 3 Without changes: When a change occurs, an error is generated, because of this type of dimension must always remain static, the ETL designer establishes in the error manager the actions that will be taken if an error is generated in the package.

The dimension's filling process doesn't show the whole steps required in an ETL package thanks to the SCD component incorporated. Source is configured by the user; however transformation and load are handled by the SCD component, due to the settings that are made inside it.

Most ETL's tools available at market allow the implementation of slowly changing dimensions; these are very useful because with just some few configuration steps three types of SCD's are created. Most SISST Data warehouse dimensions, doesn't present variations over time, catalogs have been updated just a few times. Despite of catalogs and dimension doesn't change a lot, is necessary to maintain the changes. Because of this, SCD type 2 has been implemented in all the dimensions. It's important to mention that within a certain dimension it's possible to have columns that reflect the three types of SCDs.

B. *Facts*

There is nowhere to be seen as clearly as in the fact table the ETL process, a large number of operations take place in this part of the process, first of all data quality is verified, then it is transformed in order to fulfill the data warehouse requirements and at last, it is loaded into the destination.

Extraction

SISST´s data in some cases comes from multiples sources and doesn't have the same format; SISST's data warehouse has at least two data sources:

-SISST database, CFE´s general information about accidents comes from this source (number of accidents, lost days, economic impact, etc.).

-Indicator's official information it's obtained from SIACIG (frequency, Gravity, and mortality).

Cleaning and transformation

SISST´s data warehouse transformation process consist of multiple steps, the following steps were made in order to perform the required conversions:

- Null conversion: In most cases transformations area related to null conversion, null value doesn't mean anything, this just indicates than a value was not inserted in the database, a default value must be inserted instead null value when a row hasn't a value for a column, as a best practice. A lot of integration projects made this kind of transformations. A default value was looking for each column without a value, after that, null value was replaced with this, as an example, age attribute's null value is replaced with a zero, this means "without registered age" at dimension.
- A second transformation: it's about non job related accidents and non job related transportation accidents, these are two different types of accidents, however it was decided to create a single category for the management of both. It was also created a column that substitutes them both in the fact table, depending of both kinds of accident's combination they will be a substitute for a single value.
- Data Conversion: Changing data types is a very common task in order to adapt and adjust them to fit better the data warehouse. In this step we have two possible choices, as first choice use SQL standard (the natural language for querying databases); a second option is to use a component in order to make the conversion. SISST´s data warehouse uses a component to make changes in data types. (see Fig. 4).

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

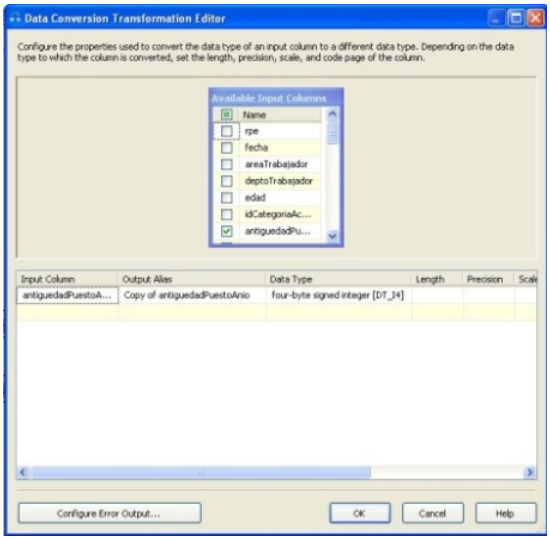

Fig. 4 Data Conversion

- Descriptions for keys: Source data table has fields that indicate the key area where an accident occurs, for SISST's data warehouse keys are not necessary, it doesn't mean anything for final users, instead of them, we need to change them for descriptions which mean much more to users. "Look Up" component was used to perform this task, "Look up" component creates a relation between data flow and other source table, and in this case "Look Up" component is used to look the description of the area stored at Areas table. This is a classic example of transformation, as shown in Fig. 5; this search was conducted for each organizational level.
- Age and seniority: The age and seniority are two special cases that require special mention. Injured workers age is storage in a source table as an integer. It's important for the final user to see these values as ranges, because of these intermediary tables were created at ODS, these tables storage the ranges required by the final user, after that substitute them in the fact table for their matching key.
- Time dimension: Time dimension is not structured explicitly within the origin table which contains the date attribute; the fact table contains the date attribute as well as bimonthly, trimester and semester attribute in order to make a simpler classification of the accident date and the day of its introduction into the system. To be able to do this, ODS was preloaded with each date and its key, this is an invert process compared to the one performed in "Descriptions for keys" transformation. This particular step is a major characteristic when a time dimension is created in a data warehouse.

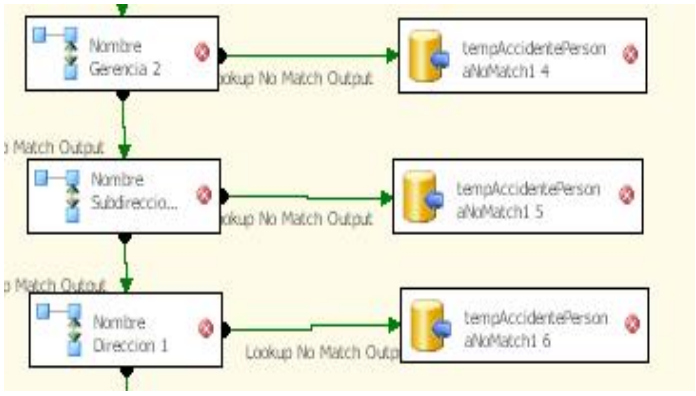

Fig. 5 *Look Up* component use in order to match the area's description

Integration

In data integration process the data source is verified in order to match the amount of rows contained in the data warehouse, if they are matching, the update and uploading process will continue, otherwise the operations specified in the error manager will occur. In SISST's case error manager will only write the mismatch rows in a text file, to analyze afterwards and to establish the necessary arrangements in order to avoid losing rows.

Update and uploading

Before uploading data into the data warehouse it is necessary to perform a task also known as updating, in this process two types of data will be verified: the ones that have been changed and the new ones. Therefore two processes will be performed:

- For data that has been changed like an SCD, once the rows that have been modified are identified will be updated as non valid, by doing this you have a time stamp for an specific date, for SISST´s data warehouse "FechaFin" column was created to store the date in which a row ceased to be valid, if this column has a null value it means that the row is still valid.
- The new ones are uploaded to the destination table.

Extraction, transformation, and updating data are the necessary steps in order to adjust the information to fit into the data warehouse requirements. After that data warehouse's upload process stars, this task can be performed with many components:

- Data mining training model
- Data reader
- OLE DB Destination

SISST's data warehouse, has a platform based on SQL Server 2008, the data uploading was made with an OLE DB component. Data´s destiny and table columns are configured as inputs for the component. The facts table's columns must be mapped in order to match de data flow columns. At 2:00 am a daily agent makes the data uploading, this task is done at SQL server by a SQL Server agent's job.

V. PACKAGE CONTAINER

Two ways can be followed after package design and deploy:

1) Deploy packages into the server to create a Job and

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

configure each package, within SQL Server Agent. It means to create a step, configure the error manager in case a package fail and organize each package. This isn't the best option, a user might find some problems with the job's creator interface, because it doesn't allow a great flexibility for error handling.

2) A better implementation in order to control the execution and error handling of packages, should be designed with a package container, this container organize execution and error handling in packages.

The main purpose of the package container is to control the execution order; is a very useful tool for error handling and also shows a graphical view about the execution and result of the execution.

As seen in Fig. 6, the Package has two package containers, the first makes all the operations needed in order to upload the ODS with data about age and seniority, the second one uploads dimensions and secondly uploads fact table in the data warehouse.

The package container design makes easier the deploy task, as mentioned before, this is not an obligatory task, however it minimize the solution's deploy time.

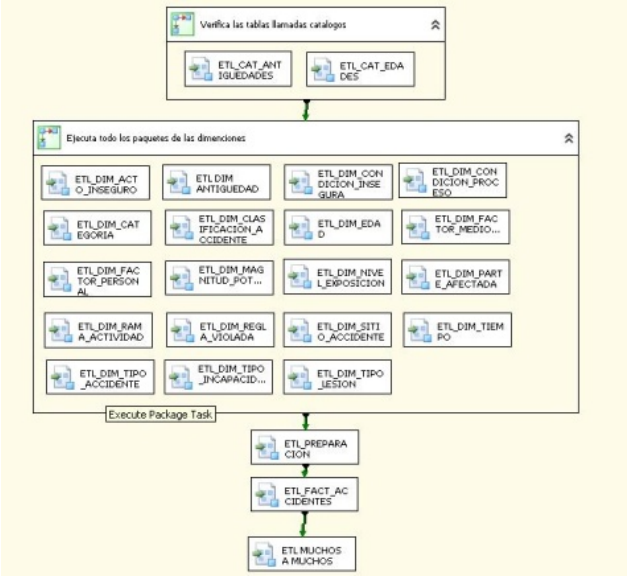

Fig. 6 Package container orders and executes a package set

VI. SOLUTION'S DEPLOY

Deployment utilities for each package are created in this last step. Before a ETL's project starts is important to identify the data sources and destinations, create them in the "Data Sources" pane and import them every time you need, as a best practice. This saves a lot of time, because it isn't necessary to create every data source in every single package. Moreover it avoids that in this step (solution's deploy step) data's sources must be configured for each package, it's only necessary to set connection strings at data's source pane.

When the project is built a manifest file is created in the folder 'bin\deployment', there is also created a copy of the different packages .dtsx and xml configuration files. In this folder there will be the following files:

- 'Configuracionde paquetes.SSISDeploymentManifest', manifest.
- Our packages dtsx
- 'Configuracion.dtsConfig', XML file previously created.

With all this the assistant for packages installation may be used

VII. CONCLUTIONS, IMPORTANT CONSIDERATIONS FOR AN ETL'S PROJECT AND FUTURE WORK

In order to create ETL's projects with high performance is important to follow a methodology without avoiding any step: extract, clean, transform, integration and update data, must be done as an automatic process for each of them.

Data profiling is an activity that always must be done before this sort of project is developed, it isn't part of the ETL's process, however it amplifies the data knowledge that will be exported, and gives the mechanisms that must be implemented in the project to clean and transform the information. This activity is so important that, if it's not done there is grave danger of failure or implementing solutions that doesn't fulfill the data warehouse's requirements.

Modularity is an indispensable requirement for an ETL project. A package that contains all the components, will be hard to read and comprehend, the error handling will be impossible, above all maintaining will be really difficult. At this point it's concluded that dividing the package in so many parts as possible is primordial.

An important matter why an ETL project fails, it's because users don't trust enough in the data warehouse, this must be taken into consideration when the ETL is designed, this is why data warehouse's snapshots must be saved, to be able to do this, a control version must be implemented in your ETL each time the updating data process marks changed or eliminated rows with the updating or elimination date. If a user doubts data warehouse's information integrity, information's veracity must be confirmed. This doesn't ensure the data quality, because it is implied through the entire ETL process.

If the tool use of ETL development allows creating package templates, these must be used as much as possible. Repetitive task must be identified and templates must be created for each one. These can be used in other packages or projects.

Nomenclature is important, dots must not be used in order to avoid problems in operating systems, and packages must have short names that reflect the usage of each package. Long names might be cut off by some tools creating mistakes in deploy process.

Data transference from the source towards its destiny, it's a task that consumes many server's resources; this is why it must be verified that in each query only the indispensable columns are imported. This helps to eliminate the additional time and improves performance.

Data sorting is an operation that consumes many resources as well, some tools have a sort component, this component consumes a lot of memory degrading package's speed and performance. As| best practice, SQL queries must

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

be used instead of sort components.

Data's sources must be set in the Project and must be shared with each package. Creating a data source for each package may cause unnecessary and tedious work to be able to deploy the solution.

For components like *join*, *look up*, etc., it's necessary to identify first the mismatching rows if there are any. This data is very useful for information integrity verifying process (amount of rows in the origin and destination must be verified), this allows to visualize the lost rows and apply improvements to the ETL packages.

If the tool allows it, use package container, this is a very valuable tool to control package execution process and error handling. It also facilitates the solution deploy process, above all it allows to understand graphically what happens with data's flow from its origin to its destination.

A final necessary step to integrate an ETL complete solution is data integrity verifying, meaning that the amount of rows in the source must be counted and compared against the amount of rows in the destination. If they are different, it means that in some part of the ETL process there were eliminated rows, this might give inconsistent data. This is a work that hasn't been automatized in SISST's packages design, however is planned to incorporate it in the future.

REFERENCES

[1] R., Kimball, *The Data Warehouse Toolkit: Practical Techniques for Building Dimensional Data Warehouses*, New York: John Wiley & Sons, 1996.

[2] I. Millet, D. H. Parente, and J. L. Fizel, *Data Warehouse Design for Ease of Data Transformation*, International Workshop on Design and Management of Data Warehouses, 2002.

[3] P. Vassiliadis, *Gulliver in the land of data warehousing: practical experiences and observations of a researcher*, International Workshop on Design and Management of Data Warehouses Stockholm, Sweden, 2000.

[4] W. H. Inmon, *Building the Data Warehouse (5th ed.)*, J Hoboken, NJ: John Wiley & Sons, 2005.

[5] R. Kimball and, J. Caserta, *The Data Warehouse ETL Toolkit: practical techniques for extracting, cleaning, conforming, and delivering data*, USA: Wiley, 2004.

ISBN: 978-988-19251-6-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)