MANAGING MULTISOURCE DATABASES: BETWEEN THEORY AND PRACTICE

(Practice Oriented)

Soumaya Ben Hassine-Guetari
A.I.D., France
sbenhassine@aid.fr

Brigitte Laboisse
A.I.D., France
blaboisse@aid.fr

Abstract: The need of managing multiple data sources in information systems is increasing and managing integrated access to information that is spread over multiple, disparate, and heterogeneous sources is being among the critical problems encountered in modern information systems. In fact, modern applications and processes used in contemporary companies often solicit distributed sets of data. These data are most of the time characterized by a high overlapping rate as referring to the same or similar entities, entailing, therefore, the necessity of consolidating it. This process of integrating and sharing scattered data is known, in the industrial domain, as the critical part of entity resolution and Master Data Management (MDM) projects, the core business of the Data Quality business unit of A.I.D..
Our article describes the best practices observed by A.I.D. after a decade of managing multisource information systems. This practice-oriented study is preceded by a theoretical overview of the numerous techniques and solutions of handling multisource information systems that underlines the gap existing between the industrial world and the academic one.

Key Words: data quality, multisource information systems, master data management, data consolidation, merge rules, data integration.

1- INTRODUCTION

The need of managing multiple sources of data in information systems is increasing; and managing integrated access to information that is spread over multiple, disparate, and heterogeneous sources is being among the most critical problems in modern information systems. In fact, contemporary companies often solicit applications requiring sets of distributed data most of them representing a high overlapping rate as referring to the same or similar real world entities.
Data integration techniques aim, therefore, at consolidating common heterogeneous data concepts spread over the underlying multisource information system and turn it into *actionable* effective knowledge [11], the key concept that is of main interest to business people.
In this context, researches in the academic field started in the eighties with approaches of logical integration avoiding conflicting situations by creating different views of multisource information systems as well as methodologies of physical integration that create a sort of "supermodel", a unique information system that resumes the overall available information, using data fusion techniques.
In the industrial world, interest in merging multisource information systems begun lately in the early 21st century and became a flourishing subject in the few previous years with the advent of numerous proprietary tools of multisource data management, especially Master Data Management (MDM).

This article describes the data consolidation strategy adopted by A.I.D., a French company which activity revolves around data. Particularly, we detail the business rules we implemented to manage and consolidate divergent and heterogeneous data.
A state of the art describing strategies and architectures of handling distributed and multisource information systems proposed in literature is firstly defined.

2- STATE OF THE ART: MULTISOURCE MANAGEMENT STRATEGIES

Nowadays, distributed development of information systems becomes increasingly common, driven by the globalization of companies and their businesses that use new information and communication technologies. In fact, management is more and more realized in a collaborative way where several partners, generally situated in distant places, participate in the elaboration of a common solution collecting data from the overall heterogeneous and autonomous local systems and repositories. Also, major marketing businesses have taken advantages from the advent of social networks as they use these latter to collect customer data trailing the ineffectiveness of the resulting distributed information system (web data, local data, other external data).
This ensuing decentralized architecture implies many inconsistencies and uncertainty problems. Moreover, having a complete overview of the available information is not always feasible when it has to deal with separate and disparate data sources. For these reasons, managing multisource databases, especially integrating multisource databases, is being among the most critical problems in modern information systems though academic researchers have dealt with multisource integration issues since the late eighties.
In this context, we may distinguish, mainly, between two strategies of integrating distributed databases [1]:

- Logical integration (also called virtual data integration) that preserves the local autonomy of each component leaving the data at the sources distributed query processing.
- Physical integration (also named materialized data integration) based on data fusion techniques

Logical integration is managed through schema mappings and recognizes two major approaches: Local As View (LAV) and Global As View (GAV), whereas physical integration generally leads to managing a data warehoused architecture. In the following we illustrate the fundaments of these tow strategies as well as their underlying techniques. We conclude this section by a discussion on the different advantages and drawbacks of such methods.

2.1- Logical integration

2.1.a- Description

The logical integration architecture represents a distributed information system (or multisource information system (MSIS)) based on **user views** that resume a set of **heterogeneous** and **autonomous** information sources. The general and referential model of the commonly adopted architecture is the **mediator-wrapper architecture** where the wrapper provides a unified data model to the mediator and where the mediator handles the communication between the sources and the system users by decomposing, in a first time, global queries into queries on the schemas of data sources, then combining and reconciling, in a second time, the multiple answers coming from the different wrappers [1]. Such architecture is represented inFigure1.

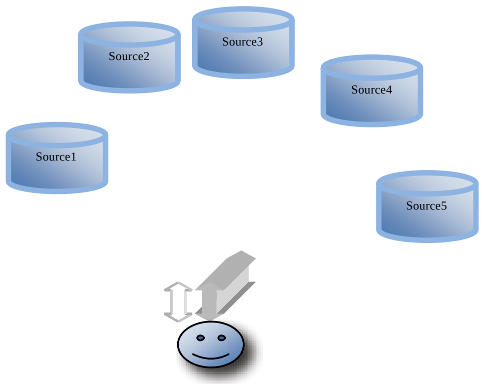

Figure1.Mediator-wrapper architecture

In fact, this mediator-based architecture handles three layers:

- The *source* layer, represented by each source and its associated wrapper. The *wrapper* plays the role of interface between the source and the mediator as it translates the queries and the queries' responses that pass through it.
- The *mediation* layer, represented by the mediator; that has in charge the transformation and integration of the information obtained from the sources, according to the requirements coming from the application layer. It offers, therefore, an integrated access through its global schema.
- The *application* layer, represented by the user; that provides the user views to the user applications through execution of queries over the mediation layer.

In real world applications, this mediator-based architecture appears in different forms and we distinguish:

- ***Federated databases***, where disparate sources need to be combined with "*approximate matches*" **preserving the local autonomy of each component**. Access to internal and external sources has to be made as if it were one logical database. [1]
  Many integration strategies has been developed in this context, known as **negotiation processes**, the majority depending on the coupling architecture where we distinguish loosely coupled systems and tightly coupled ones. On one hand, in loosely coupled systems, every user has to carry out his own integration schema and negotiation is made by establishing bilateral agreements between the different sources through contracts. On the other hand, in tightly coupled systems, negotiation is done by means of a global integration layer (managed by a global or federated administrator) to combine all local schemas into a global one. In this latter configuration, federation administrator has to converse with each source's administrator in order to reach an agreement about the contents of the export schemas and operations allowed on them [14].
- ***Cooperative information systems***, where a cooperation layer provides a gateway between the overall multiple sources and offers mapping between the data schema in each.

The difference between cooperative information systems and the federated ones lies, mainly, in the definition of the global conceptual schema. In fact, in federated information systems, the global schema defines the conceptual view of the entire database, while, in the cooperative solution, it represents a collection of some of the local databases that each local database system is willing to share out. Moreover, according to Mecella [13], collaborative information systems are characterized by high data replication, i.e., different copies of the same data are stored by different organizations. Also, they are designed according to a service-based approach where cooperation is obtained by sharing and integrating services across networks. This is illustrated by the researches of Tari et al. [17] that propose an agent-based architecture with coordination agents to identify and delegate the subparts of users query to appropriate agents of the task layer, specialized agents that use the agents of the database layer to access the required information to process the different sub-parts of the global request and wrapper-coordination agent that offer a reliable cooperative environment for information sharing and handles security services with different levels of autonomy.

Nowadays, many evolutions of logically-integrated architectures have been developed. In 2008, Dynamic Distributed Federated Databases [3] are defined as a network supporting ad-hoc operations providing efficient access to distributed sources of data and which advantage especially appears when the applications requiring the data ignore the location of the data in the underlying network. This is done through the use of mechanism called “store locally, query anywhere”. In fact, this mechanism provides for global access to data from any vertex (node) in the database network. Data is stored in local database tables at any vertex in the network, and it is accessible from any other vertex using Structured Query Language (SQL) like queries and distributed stored procedures-like processes. Then, the query propagates through the network and result sets are returned to the querying vertex.

Beyond their denominations, the real difference between these architectures lies in the integration schemas. As we enounced in the introductory paragraph of this section, we may distinguish between local as view technique, the global as view technique and other hybrid strategies. These techniques are described in the following paragraph.

2.1.b- Logical integration techniques

Generally, logical integration uses the three-layered architecture based on the source layer, the mediator layer and the application layer. We distinguish two major logical integration techniques: LAV and GAV. In the LAV approach (where local refers to the local sources), sources act as a view over the mediated schema, whereas in the GAV one (where global refers to the global schema) the mediated schema acts as a view over source relations.

Independently from the adopted integration method, the query processing requires in the both cases a reformulation step. For instance, the Information Maniford (IM), a LAV integration system, uses a plan generator for query reformulation used to prune irrelevant sources, split query into subgoals, generate conjunctive query plans and find executable ordering of subgoals. For The Stanford-IBM Manager of Multiple Information Sources (TSIMMIS), a GAV integration system, the query reformulation is straightforward as it uses wrappers that act as database drivers and handle the communication between the mediator global schema and the data sources [4, 10].

Each of the methodologies has advantages and drawbacks. In fact, for LAV systems, adding new sources is straightforward as it only needs sources specification. However, it becomes trickier when it has to reformulate users’ queries. In the contrary, query reformulation is far simpler for GAV systems but far more complicated when it has to deal with adding, removing or modifying sources. As a conclusion, LAV fit for dynamic, distributed systems while GAV systems are recommended for static and centralized systems.

Beyond these two integration systems, hybrid integration techniques have been proposed in literature. For instance, in 2003, Mc. Brien and Poulovassilis [12] define the Both-As-View (BAV) methodology that takes the advantages of the two previously defined integration schemas: the LAV schema and the GAV one as it is based on the use of reversible schema transformation sequences. Moreover, semantic relationships between the contents of data sources and relations in the mediated schema are specified using logical language.

Other researches worked on the enhancement of such a hybrid approach. In fact, in 2010, Rizopoulos et al. [15] defined a schema matching/mapping approach that takes into consideration the uncertainty affecting compatibility mappings as well as semantic mappings that is a highly intelligent process, and extremely tough and time consuming even for an expert user because of being a highly subjective task. In the proposed approach based on the use of the HDM model (Hypergraph Data Model- a common data model resolving model heterogeneity problems), uncertainty is managed through ranking scores that rank the possible semantic mappings for each pair of objects based on their likelihood. By ranking the possible semantic mappings, the ones that are more likely can be investigated first and they can be used to produce integrated schemas that provide the correct query answers.

2.2- Physical integration

2.2.a- Description

The difference between logical integration systems and physical ones lies in the fact that the unified view of data encountered in logical systems is materialized, in physical systems, in a data warehouse or a centralized database as illustrated in Figure2.

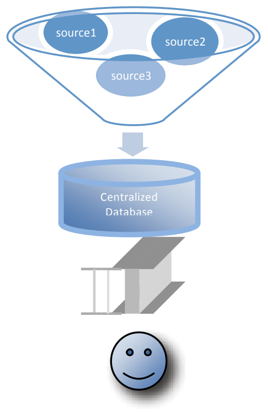

Figure2. Physical integration

According to Bleiholder and Naumann [2], physical integration (also called **data fusion**) is based on a three-step process:

- ***data transformation***: used to present the user query results in a single unified schema. Two approaches are common to bridge heterogeneity and thus specify data transformation: schema integration and schema mapping. Schema integration is driven by the desire to integrate a known set of data sources. Schema integration regards the individual schemata and tries to generate a new schema that is complete and in the same time minimal, correct and understandable with respect to the source schemata. Schema mapping, assumes a given target schema driven by the need to include a set of sources in a given integrated information system.
  These techniques are also encountered in logical integration when conceiving a GAV schema.
- ***duplicate detection***, that is technically the same as record linkage, entity resolution, object identification and reference reconciliation [8]. The goal of this step is to identify multiple representations of the same real-world object: the basic input to data fusion.
  This step is required in both integration methodologies (LAV and GAV).
- ***complete/concise data integration***: As reported in [2], data integration has two broad goals: increasing the completeness and assuring, at the same time, the conciseness of the overall available data. An increase in completeness is achieved by adding more data sources (more objects, more attributes describing objects) to the centralized system. An increase in conciseness is achieved by removing redundant data, by fusing duplicate entries and merging common attributes into one.

Just as in logical integration, physical integration trailed many algorithms and strategies we intend to develop in the following paragraph.

2.2.b- Physical integration techniques

In 2009, Dong and Naumann set out some of the techniques used for resolving conflicting integration problems and they distinguish between **instance-based techniques** where conflicts are resolved on the value level; and **metadata-based** techniques where source quality indicators such as freshness and reliability are determinative [8]. In this paragraph, we will use this classification to describe physical integration techniques.

In 2001, Wang et al. [18] define the ploygen model that focuses on analyzing data provenance in order to appreciate its underlying credibility. In fact, the polygen model is among the first metadata-based models that studies heterogeneous database systems from the multiple source perspective as it attempts at defining data credibility based on two important views: data source knowledge and intermediate data source knowledge. Thus, by resolving the data source tagging (credibility of data given knowledge of its source) and the intermediate source tagging problems (credibility of data given knowledge of its source as well as knowledge of the intermediate sources that helped in composing the data), the polygen model addresses critical data provenance issues and, therefore, entity resolution problems where the accuracy of a data value is tightly related to the credibility of its provenance.
The same strategy was used, in 2004, by Cholvy [5] who based his fusion process to information quality evaluation so as to build an up-to-date and correct centralized system. This evaluation process uses the STANdardization AGreements 2022 (STANAG 2022) model that assesses the pair of source quality indicators: *source reliability* and *information credibility*, where a source is considered reliable when it is tried and trusted and a data value is considered credible when the reported information is "confirmed by other sources". Evaluation of both source reliability and information credibility is based, therefore, on the historical use of the underlying sources. Given this assessment rule, new sources are, unfortunately, considered unreliable and non-credible.
Besides, in [16], Talburt assimilates the accuracy of data values to the accuracy of the sources they come from. For this sake, he suggests to use a naïve estimation approach based on the assessment of the accuracy of a data sample originally extracted from these sources. Though simple, the efficiency of this approach strongly depends on the representativeness of the extracted sample as well as the robustness of the extrapolation algorithm, more sophisticated accuracy estimation rules have, then, to be suggested. For that sake, Talburt, proposes the use of supervised learning techniques (a machine learning technique) such as neural networks, fuzzy computing, evolutionary computing, classification algorithms as well as support vector machines. He also suggests the use of multiple attributes analysis where a determinate attribute predicts the value of another one giving the dependency relation it might exist between them.
Beyond the structured data, metadata-based consistency resolution techniques were also proposed in the case of web data (unstructured data). In fact, scoring functions were suggested to prefer a web source to another. Some of these scoring functions are based on the frequency of the answers appearing in the web results as well as the originality of pages reporting the answer. [19]
Moreover, a more sophisticated approach is defined in 2009 by Dong et al. [7] who propose the use of a probabilistic model of **truth discovery** to resolve conflicts when integrating data from multiple **web sources**. This model is based on the analysis of the trustworthiness of these integrated values by evaluating the *accuracy of their underlying sources*, the *dependence* that may exist among them and *confidence* of the integrated values. In fact, the authors use Bayesian rules to compute the probability of a data value to be true and use this probability to compute sources accuracy. Applied to real world data, this model proves its robustness as it detects violations of assumptions, indirect source copying, in addition to preventing from falsifications.
Most recently, with the advent of social networks and their use to collecting business information, much of the current researches focus on association analysis in order to learn about data provenances. Such associations are modeled in [16] as network graphs in which references are represented as nodes and the edges between the nodes represent associations between the references.

On the other hand, many instance-based consistency-resolution approaches have been defined in the literature. We may, therefore, distinguish [2]:

- ***Conflict ignorance*** where no decision about conflicts among data is made. For instance, we can quote the "*pass it on*" strategy and the "*consider all possibilities*" one.

The "pass it on" strategy consists in leaving the choice decision to the user. In the same manner, the "consider all possibilities" strategy consists in enumerating all eventualities and giving user the possibility to choose the preferred alternative among the overall conflicting values.

- ***Conflict avoidance strategy*** that handles inconsistencies without resolving conflicts. This method is based on two strategies:
    - *instance-based strategy*, does not take decision and is based on the "*take information*" and "*no gossiping*" principle. "Take information" considers filled information (not null) and leaves aside null ones filtering, therefore, unnecessary values. Similarly, "no gossiping" considers only consistent data for decision making leaving aside all inconsistent ones.
    - *Metadata-based strategy*, where metadata are considered when taking a decision based on the "*Trust your friends*" principle as a given source is favored to the others given criteria such as price, reliability, amount of data, or other quality criteria.
- ***Conflict resolution*** is the strategy that resolves conflicts among integrated information. We may distinguish between:
    - *Decision strategy*, that resolves conflicts by choosing a value among the overall conflicting values using **data lineage techniques** (that analyze data provenance).
    - *Mediating strategy*, based on **compromise algorithms** and other **recency-based** ones and may, therefore, choose values that do not necessarily exist among the conflicting values. In fact, it is based on methods such as "*meet in the middle*" that do not prefer one value over the other but rather computes a new value that is as close as possible to all available values. For instance, instead of choosing between the values 30 and 40 designating the "Number of employees" of a company, we take the average figure which is 35. This method cannot, however, be applied to conflicting values when they represent a large gap. It is obvious that the "age" value 54 handling the conflicting values 9 and 99 given the "meet-in-the-middle" strategy is not a satisfying solution.

      Another method is the "*keep up to date*" strategy that uses the most recent value and requires some additional time-stamp information about the recency.

In this paragraph, we defined instance-based entity resolution techniques as well as metadata-based ones. However, entity resolution is not only encountered when merging heterogeneous data values, it may also concern heterogeneous model matching. In this context, Kolovos et al. [9] describe a rule-based merging model based on the EML language (Epsilon Merging Language) that attempts to merge heterogeneous models of diverse metamodels and technologies. This EML-based model is grounded, as the majority of entity resolution processes, on three steps:

1. Comparison and conformance step, where each match-rule can compare pairs of instances of two specific meta-classes and decide if they match and conform to each other.
2. Merging step, based on two activities: merge and transform, where
    a. Elements, that have been identified as "can be matched" in the previous conformance step, are merged into a sequence of model elements in the target model; and
    b. a selection of the elements for which a match has not been found in the opposite model are transformed into equivalent elements of the target metamodel.
3. Reconciliation step, where the target model, after merging the different models, is restructured to fit users' requirements. In case of inconsistencies (more than one rule are found to apply to a pair of objects), the execution process stops and the decision is entrusted to the user.

This solution has the advantage of generating a trustful target model with the drawback of overburdening the user with a set of consistency-resolving cases that is tricky to handle especially in big organizations when tens (even hundreds) of models are supposed to be merged.

2.3- Discussion

The previous paragraph illustrated the existing techniques and methodologies of managing and integrating scattered data in multisource information systems. These approaches, while efficient in theory, are impractical in real world conditions as they are either idealistic or too sophisticated. Moreover, the proposed approaches are either instance-based or metadata-based. The analysis of the correctness of a data value only based on metadata source information is dangerous as data sources are of different quality and "accurate sources can make mistakes as well" [8]. In the same manner, managing conflicts using only instance-based approaches may be misguiding such as correct but out-dated values.

In fact, what is used in the industrial domain is more a "collection of data management best practices associated with both the technical oversight and the governance requirements" of the underlying project.
Thus, MDM techniques are more generalized than logical and physical integration approaches as it overtakes the data fusion function to deal, among others, with "facilitating the sharing of commonly used master data concepts and orchestrating the management of business applications, information management methods and data management tools". It is defined by Loshin [11] as a program, more than an application or a project that is intended to provide consistent views of the overall entities used in the underlying information system which is essentially consisting of:

- a master data repository, that is a collection of components used to manage multiple aspects of what eventually is managed as master data, for instance, reference data, metadata, master data models, business rules, hierarchies and relationships between master data entities, etc;
- a master data services, such as integration and consolidation services, data publication and data access, data quality and cleansing functions, access control and metadata management;
- and the associated governance services as incident reporting and incident tracking activities, notification and alert management, data browsing, data profiling and assessment, history and log management, privacy policy management, stewardship role management and governance policy management.

In the following section, we illustrate the integration process adopted by A.I.D..

3- MULTISOURCE DATA MANAGEMENT PROGRAM AT A.I.D.

3.1- The context

As defined in the discussion above, MDM projects deal with merging data coming from multiple sources: web data, suppliers' files (prospects information or enrichment data as phone number),etc.
Our main concern in such activities consists in conflicts handling when a unique alternative is to be chosen among a set of concurrent ones. Actually, we were constantly faced, in the projects we dealt with in A.I.D., to choose between a set of concurrent addresses related to the same real person, or between different birth dates qualifying that same person.
In this section, we describe how A.I.D. manages multisource data integration. The approach we adopt privileges physical integration commonly known as data fusion to logical integration in order to conceive a centralized global system off the shelf that avoids on-line conflicts handling. These conflicts are, indeed, managed off-line through priority rules we intend to describe in the following section.

3.2- Who is A.I.D.?

A.I.D. is a French company, part of Omnicom Group, which activity revolves around data as it deals with database auditing activities as well as data mining and process optimization issues. It comprises a data quality business unit which mission is to help companies succeed in their MDM projects using a strategy based on data discovery and data knowledge.

3.3.-The need for a MDM program

After a decade of multisource databases management, some statements came into evidence.

- First, **data are constantly evolving** entailing the improvement or the damage of their inherent quality and implying, therefore, a continuous amendment of the associated merge or integration rules. These merge rules have, then, to be flexible to fit with that constant evolution of data. In the context of prospect files integration for instance, a change of the quality of the turnover information, provided by a supplier *A*, may not involve a whole evolution of the integration process; instead, **merge rules have to be set as input parameters** to that process.
- Second, CRM data, which represent an important ratio of our data, often show **conflicting situations** with different concurrent data values describing the same customer information. These conflicts have to be managed, for instance, when selecting prospects information in the context of a marketing campaign; and marketers (even brokers) don't have the ability to choose the best values on the fly, during the targeting step of the campaign realization. For that reason, a "*consolidated record*", also named "*golden record*" or "*survivor record*", has to be computed and defined as a "single source of truth" among the available concurrent alternatives.

Notice that the "golden record" concept is set in [11] and designate, actually, a by-product of data consolidation that is the materialization of the alternative that has the best "quality and correctness". It implies that a process of data assessment has been carried out to measure the accuracy of the overall possibilities. This assessment is, however, hard to undertake when handling databases of millions of records; and appraising the quality of each individual record seems to be infeasible, especially when we have to deal with customer information which accuracy is difficult to infer. Our methodology seeks, therefore, for a "record consolidation" more than a "golden record".
In the following paragraph, we describe the integration methodology adopted in A.I.D. which is based on a physical integration architecture realized through a data fusion process. We define, afterwards, quality parameters we use in order to handle conflicting alternatives named *blocks*.

3.4- In block or single attributes?

A block is defined by a set attributes and what we name "*attribute*" is not only but the database concept designating a column. For instance, if we have a table *Person*, an attribute is a column of the table *Person*, the phone number for example.

We distinguish 2 kinds of attributes:

- linked attributes: for instance, the physical address. An address is often composed by multiple attributes as *address1* (the line name), *address2* (the number), *zip code*, *city*, *country*. All these attributes are dependent: no sense to take the *address2* from a record *A* and the *zip code* of a record *B* to get the consolidated record (except in specific cases where *A* and *B* describe the same address and are more or less equivalent with minor variations). In this case of inter-dependent attributes, a "*functional block*" is associated to them and the inherent merge rules describe the manipulation of that functional block (i.e. the manipulation of the overall concerned attributes).
- independent or *single* attributes.

Once "*functional blocks*" are distinguished (i.e. attributes and blocks), the merge process can set off.

3.5- Data quality attributes

At A.I.D., we don't have a miracle key to decide, in case of divergent blocks, which one is correct. So, we created merge rules based on assessing the global accuracy of the source (using a naïve assessment of representative samples' accuracy [16]). Accuracy is not the only key parameter we used; dimensions such as freshness and feedback use have been integrated right from the start.[6]
For each functional block, or single attribute, the merge rules rely on four data quality attributes:

- **Accuracy of the Source**: file supplier, web, or loyalty card, etc.
- **Date**: the most difficult measure to obtain as it is more often skewed by information such as integration date or delivery date, while it has to be related to the date at which the event creating the underlying entity happened. For example, the date where a company's number of employees changed instead of the balance sheet publication date. This information is mainly critical for volatile attributes (related to the shelf life of the attribute) such as the email or the number of employees in a given company.
- **Processing code**: control code, or accuracy assessment return code. For instance, address normalization return code or email checking or coherence between the number of employees at company (SIREN level) or site level (SIRET level)[1].
- **Usage code or feedback use** (when it's available for one source): for instance, for an email, we may distinguish between:
  - Tested (emailed) and opened
  - Tested (emailed) and hard bounce
  - Tested with no feedback (no bounces, not opened)
  - Not tested (not emailed)

3.6- Parameters

To set up the priorities and to manipulate the data quality attributes for each functional block, we define the following parameters (cf. Figure3).

Bloc	Name of the functional block
Origin	Source of data
Processing code	Processing code or usage code
Obsolescence	The obsolescence of the data is manipulated as a score.
Honey period	The next days after an update are considered as a 'honey period' during which no other data can be so reliable to have the priority.
Limit period	Even if the source is reliable, after n days, the data is no longer valid
Score	Priority or score. The best priority is 0, the least 99

Figure3: blocks' parameters

3.7- How does it work?

Let's take an example. We have 3 sources of data:

- Source A: a CRM system, where the data is updated by salesmen
- Source B: an official data supplier which we call *ALPHA* for confidentiality purposes
- Source C: a private data supplier which we call *BETA*

We want to put in place the priority rules for the attribute "*Number of employees*". "*Number of employees*" defines the number of the site employees (SIRET level) having an open-ended contract in the current year. However, this information is often inaccurate as:

[1]SIREN and SIRET are identifiers used to respectively describe French companies and their underlying sites.

- Information delivered from salesmen are not imperatively the most reliable, nor the most accurate. Data values are not necessarily false; often, they do not respect the definition of the attribute “Number of employees” set above. In fact, delivered information is often relative to the company instead of the site number of employees and includes, sometimes, temporary contracted employees.
- Information supplied by salesmen has to be committed in the system even if this information is not valid. In fact, the attribute “*Number of employees*” is crucial in marketing services as business activities and units are divided given that information into small businesses, mid markets or big accounts. Thus, the minus change in the attribute value may imply the transfer of a prospect from a business unit to another.
- “Number of employees” is an information that may provided by the official supplier: *supplier ALPHA*. In fact, *supplier ALPHA* may gather that information from two sources. The first source is the official obligatory declarations (ADSI- Annual Declaration of Social Information) made by the company itself in the penultimate year (year N-2). The second source is surveys made by the files suppliers. However, the provenance information is never mentioned nor delivered with the data.
- A second source of “Number of employees” attribute is the private supplier: *supplier BETA*. Information gathered by *supplier BETA* may come from:
  - the same *Supplier ALPHA* , which is the official provider
  - balance sheets published by the companies
  - private surveys made on targeted samples

  Like *supplier ALPHA*, *supplier BETA* provides no provenance information.

To evaluate the accuracy of information provided by the suppliers *ALPHA* and *BETA*, we compared the values of the number of employees on common records. In fact, 85% of the records had the same value; while 15% of the records had different values, where 8% could trigger the attribution of a different business unit.
Examples of received data are described in Figure4 below.

Entity	Source A		Source B		Source C	
	CRM		Official supplier (ALPHA)		Private supplier (BETA)	
	Value	Date	Value	Date	Value	Date
1			20	2010	25	NA
2			2	2010		
3	12	09/03/2011	10	2010	9	NA
4	39	04/11/2010	18	2010		
5			187	2008	300	NA

Figure4 – Example of received data

The merge parameters which we set up are summarized in Figure5.

- Processing code: the only control put in place was the coherence between the number of employees at the site and at the company level.
  If nb_employees_site > nb_employees_company, then processing_code = KO, else OK. If the values are coherent or if one value is null, the processing code is OK
- Obsolescence: for each source, we consider a general obsolescence of 3 months. However, when the source is *supplier ALPHA* the obsolescence is of 6 months. The reason of this setting is technical: *Supplier BETA* and CRM give no update dates, *Supplier ALPHA* gives efficient update date, but has a delay of 3 months to provide the data. For instance, if you are in march 2011, by default the date for *BETA* is march 2011, the date given by *ALPHA* is, however, January 2011. The obsolescence score evolves byreducing1 point of priority every 3 (or 6) months. For instance,

the *supplier BETA* with a processing code OK will start with a priority 3that will switch to priority 4 after 3 months, 5 after 6 months, etc.

- Honey period: only for the CRM source, and to insure that the data coming from the salesman is selected whatever the other sources information, a honey period of 6 months is set up. The consequence is during the 6 months after the observation date, the CRM source is forced with a priority 0, which is the best priority. Six months later, the priority is back to the standard priority of the source (2 for CRM).
- Limit period: after 24 months, data become obsolete and priority level is set-up to 99 (the least one)

Source	Processing code	Obsolescence	Honey period	Limit period	Priority
CRM	*	3	6	24	2
ALPHA	OK	6	-	24	1
ALPHA	KO	6	-	24	99
BETA	OK	3	-	24	3
BETA	KO	3	-	24	99

Figure5. Merge parameters

Let's now proceed with applying the priority rules. We consider a consolidated record provided by *supplier ALPHA* with the following quality attributes: ***date*** is equal to *December 2010*, ***processing code*** is *OK* and ***priority level*** is *1*. Priority starts at level 1, decreases by one point each semester to reach the level 4.

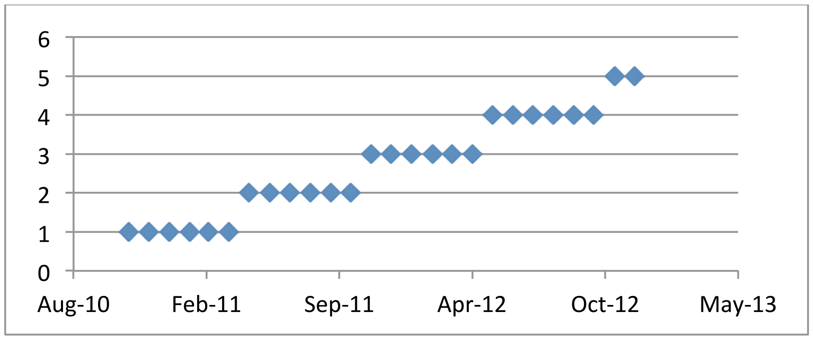

Figure6. Priority v1

Let's now integrate 4 flows:

- Flow 1: in May 2011, the CRM source gives a divergent data. The priority level is 2, compared to the consolidated record for which the current priority is 2. For the same priority, the latest date wins, allowing to select CRM. The lines 'consolidated record v2' and 'priority v2' give the new projection in the future of the consolidated record. Let's observe that we have 6 months from May to October 2011 where the consolidated record has the best priority 0, which means that whatever the external source, the value will not be changed.

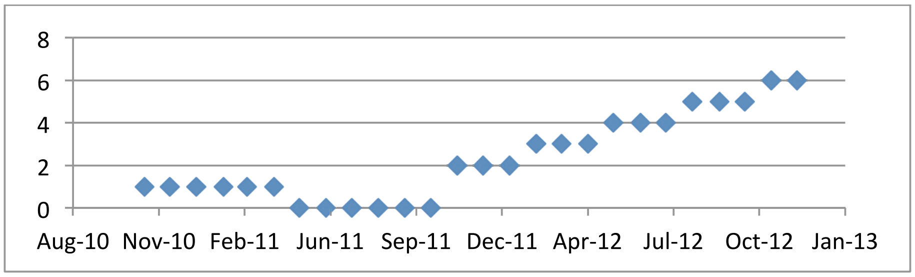

Figure7. Priority v2

- Flow 2: in August 2011, *supplier BETA* gives a new data. That data is processed as "*august data*" even if *supplier BETA* does not give any tractability in term of date. The priority level of *BETA* is 3, compared to the current 0 of the consolidated record: the update is ignored. Lines "consolidated record v3" and "priority v3" are identical to the version V2.

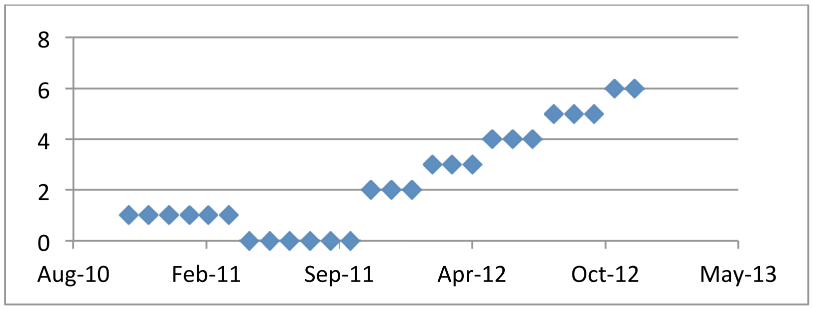

Figure8. Priority v3

- Flow 3: in June 2012, *supplier BETA* came with a new data. The priority level of the consolidated data is now 4 compared to 3 for the external source: the external source wins. The result is Priority v4 for the consolidated record :

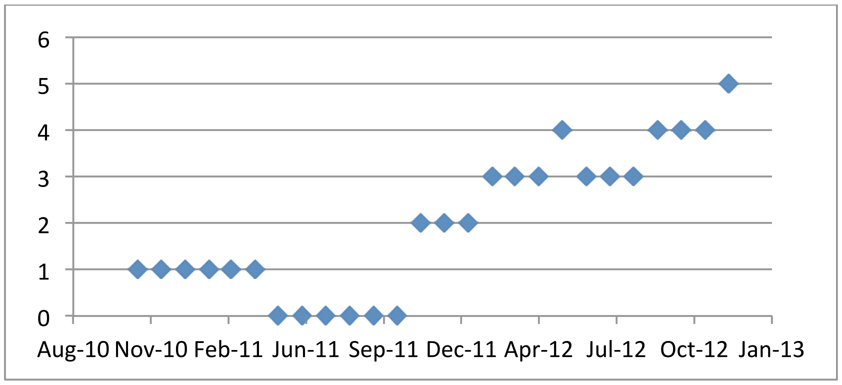

Figure9. Priority v4

- Flow 4: in July 2012, *supplier ALPHA* provides a new data, but time stamped to December 2011. So, the priority level is evaluated to :

  priority = 1+ number of months between (December 2010 and July 2012) / obsolescence
  = 1 + 7 / 6 = 1 + 1 = 2

  At the same time, the consolidated record has the priority 3, which triggers the update by *ALPHA*.

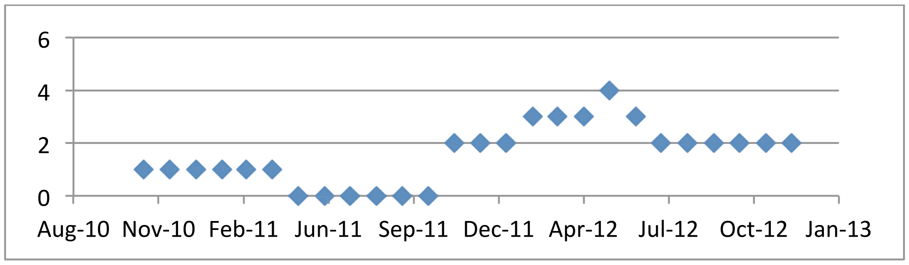

Figure10. Priority v5

3.8- Does it work?

Our ten-year experience in data merge comes out with the following points:

1- The merger rules described above are the first stone of any integration project. Other more sophisticated rules may eventually be added if needed.

2- The rules set-up needs good data sources knowledge. For this reason, additional surveys may be undertaken to assess the accuracy of the underlying data.

3- It is important to take into account the usage feedback and integrate it as a key element in the migration project

4- Priority rules depend on the underlying data. As system data is in continuous change, priority rules need, also, to follow this evolution and re-evaluated if necessary.

5- When building the "consolidated record", all concurrent alternatives given in input may not be taken into consideration and a filtering out algorithm may be used to take off, for instance, null values or inconsistent ones.

4- CONCLUSION

This paper illustrated the entity resolution approach adopted by A.I.D.. More especially, it describes the adopted merge rules that serve to compute a "reliable consolidated record", the most important challenge in MDM projects. These merge rules have to be quite simple, intuitive and comprehensible to users. They also need to be standardized in order to assure their interoperability with the overall autonomous sub-systems of the global MSIS.

However, given the state of the art and the industrial example, we can notice the gap that exists between theory and practice. In fact, while researches focus on data provenance analysis with complicated techniques based on probabilistic models and other networked dynamically evolving graphs, industrials struggle to find the data provider identity; while researches base their merging solution on data recency models, industrials labour to build correct, not null information update knowledge; finally, while researchers propose idealistic solutions, industrials introduce customized but less rigorous approaches due to the lack of theoretical applicable suggestions. As a conclusion, a gap between these two communities is created making industrials no longer profit from the theoretical findings.

So, dear professors and doctors, what do you suggest to improve industrials' merge rules? Could you help them automatically detect the evolution of their sources' accuracy? How can industrials set up an obsolescence period?, and many other questions that bring us to the conclusion that a bridge between these two worlds (industrial and theory) has to be built and strengthened.

REFERENCES

[1] Batini, C. and Scannapieco, M., *Data Quality: Concepts, Methodologies and Techniques*, Springer Verlag Berlin and Heidelberg GmbH & Co. K publisher, 2006.

[2] Bleiholder, J., and Naumann F., *Data Fusion*, ACM Computing Surveys (CSUR), 41 (1) 2008.

[3] Bent, G., Dantressangle, P., Vyvyan, D., Mowshowitz A. and Mitsou V., *A dynamic distributed federated database*, 2nd Annual Conference of ITA, Imperial College, London, 2008.

[4] Chawathe, S., Garcia-Molina, H., Hammer, J., Ireland, K., Papakonstantinou, Y., Ullman, J. and Widom J., *The TSIMMIS Project: Integration of Heterogeneous Information Sources*, 10th Meeting of the Information Processing Society of Japan, 1994, pp.7-18.

[5] Cholvy L., *Information evaluation in fusion: a case study*, Information Processing and Management of Uncertainty, 2004.

[6] Clement, D., Ben Hassine-Guetari, S. and Laboisse B., *Data Quality as a key success factor for Migration Projects*, 15th International Conference on Information Quality, 2010.

[7] Dong X.L., Berti-Equille, L. and Srivastava D., *Integrating conflicting data: the role of source dependence*, VLDB Endowment 2(1), 2009.

[8] Dong X.L. and Naumann F., *Data Fusion: Resolving Data Conflicts for Integration*, VLDB Endowment 2(2), 2009.

[9] Kolovos, D.S., Paige, R.F. and Polack, F.A.C., *Merging Models with the Epsilon Merging Language (EML), Model Driven Engineering Languages and Systems* In Model Driven Engineering Languages and Systems , 4199, 2006), pp. 215-229.

[10] Levy A.Y., *The Information Manifold approach to data integration*, IEEE Intelligent Systems, 13, 1998, pp.12-16.

[11] Loshin, D., *The practitioner's guide to data quality improvement*, Knowledge Integrity Inc., Morgan Kaufmann publisher, 2011.

[12] Mc. Brien, P. and Poulovassilis A., *Data integration by bi-directional schema transformation rules*, 19th International Conference on Data Engineering 2003, pp. 227-238.

[13] Mecella, M., Scannapieco, M., Virgillito, A., Baldoni, R., Catarci, T. and Batini C., *Managing data quality in cooperative information systems*, 10th International Conference on Cooperative Information Systems, 2002.

[14] Olivia M. and Saltor F., *A negotiation process approach for building federated databases*, 10th ERCIM Database Research Group Workshop on Heterogeneous Information Management, 1996.

[15] Rizopoulos N., *Schema Matching and Schema Merging based on Uncertain Semantic Mappings*, PhD Thesis, Imperial College London, 2010.

[16] Talburt J.R., *Entity resolution and information quality*, Morgan Kaufmann Publishers, 2011.

[17] Tari, Z., Zalavsky, A. and Savnik, I., *Supporting cooperative databases with distributed objects*, Parallel and Distributed Systems: Theory and Applications, J.L. Aguilar Castro publisher, 1998.

[18] Wang, R.Y., Ziad, M. and Lee, Y.W., *Data Quality*, Kluwer academic Publishers, 2001, pp19-32.

[19] Wu, M. and Marian, A., *A framework for corroborating answers from multiple web sources*, Information Systems Journal, 36(2),2011, pp 431-449.