6

Data Quality Issues in Data Integration Systems

6.1 Introduction

In distributed environments, data sources are typically characterized by various kinds of heterogeneities that can be generally classified into (i) technological heterogeneities, (ii) schema heterogeneities and (iii) instance level heterogeneities. *Technological heterogeneities* are due to the use of products by different vendors, employed at various layers of an information and communication infrastructure. An example of technological heterogeneity is the usage of two different relational database management systems like IBM's DB2 vs. Microsoft's SQLServer. *Schema heterogeneities* are principally caused by the use of (i) different data models, such as one source that adopts the relational data model and a different source that adopts the XML data model, and (ii) different data representations, such as one source that stores addresses as one single field and another source that stores addresses with separate fields for street, civic number, and city. *Instance-level heterogeneities* are caused by different, conflicting data values provided by distinct sources for the same objects. This type of heterogeneity can be caused by quality errors, such as accuracy, completeness, currency, and consistency errors; such errors may result, for instance, from independent processes that feed the different data sources.

Today, there are many examples of scenarios in which data residing at different sources must be accessed in a unified way, overcoming such heterogeneities. *Data integration* is a major research and business area that has the main purpose of allowing a user to access data stored by heterogeneous data sources through the presentation of a unified view of this data. Though data integration must face all the types of heterogeneities listed above, in this chapter we focus particularly on instance-level heterogeneities, where data quality issues become very significant. Indeed, instance-level heterogeneities can strongly affect query processing in data integration systems. Specifically, the query processing activity can be performed by considering that different data sources may exhibit different quality levels for the data. Hence, answering algorithms can be executed to provide the optimal quality results for the final user. We will describe some approaches to such *quality-driven query processing*. Furthermore, when collecting data as answers to queries, possible conflicts must be solved, by means of a specific *instance-level conflict resolution* activity; otherwise, the whole integration process cannot be correctly terminated.

Quality-driven query processing and instance-level conflict resolution can be seen as two complementary approaches that deal with instance-level heterogeneities. Specifically, it is possible to consider

1. only-quality driven query processing (without conflict resolution);
2. only conflict resolution (without quality-driven query processing);
3. both approaches used complementarily.

Quality-driven query processing modifies the query answering semantics in order to take into account varying quality of source data. It can assume (case 1) that instance-level conflicts are not solved, but metadata are available in the system to return the best quality answer (see [142]). Instance-level conflict resolution can focus on solving conflicts between sources independently of the query processing (case 2), for example by operating not at query time but at a different phase of the data integration process, such as the population of a data warehouse (see [141]). Alternatively (case 3), conflicts resolution techniques can be performed at query-time, within the quality-driven query answering process itself (see [175]).

In this chapter, first we describe some basic concepts on data integration systems (Section 6.2). Then, we provide an overview of existing proposals to deal with quality-driven query processing (Section 6.3) and instance-level conflict resolution (Section 6.4). Finally, we give some insights into theoretical proposals to address inconsistent query answering in data integration systems (Section 6.5).

6.2 Generalities on Data Integration Systems

Two main approaches to data integration can be identified, based on the actual location of data stored by sources to be integrated

- *virtual data integration*, where the unified view is virtual and data reside only at sources. A reference architecture for virtual data integration is the mediator-wrapper architecture [209]; and
- *materialized data integration*, where the (unified view of) data is materialized, for instance, in a data warehouse.

In this chapter, we will refer mainly to *virtual* data integration. When describing quality-driven query processing we will essentially focus only on virtual data integration systems. In contrast, the concepts related to instance-level conflict resolution techniques can be applied in both virtual and materialized data integration scenarios.

In the following section, we will describe the major features of a virtual data integration system. As already discussed in the introduction, data integration is the problem of combining data residing in different sources, providing the user with a unified view of this data, called *global schema*. A data integration system (DIS) is composed of three elements: (i) a *global schema*; (ii) a set of *source schemas*, including schemas of all sources; and (iii) a *mapping* between the global schema and the source schemas, which specifies the relationships between the concepts expressed in the global schema and the concepts in the source schemas.

Virtual data integration typically assumes a mediator-wrapper architecture, depicted in Figure 6.1. Wrappers have the main task of providing a uniform data model to the mediator. The mediator has the task of decomposing the global query into queries on the schemas of data sources. Furthermore, the mediator must combine and reconcile the multiple answers coming from wrappers of local data sources.

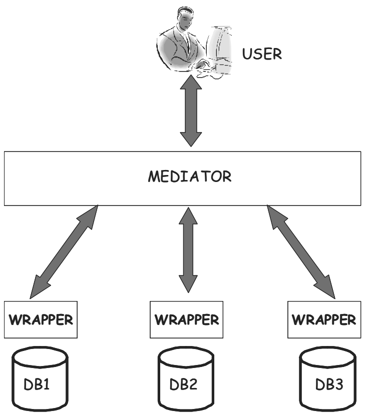


Fig. 6.1. Mediator-wrapper architecture

Two basic approaches have been proposed to specify the mapping [116]. The first approach, called *global-as-view* (GAV), requires the global schema to be expressed in terms of queries (or views) over the data sources. The second approach, called *local-as-view* (LAV), requires each data source to be expressed in terms of queries over the global schema. A third approach is called *global-local-as-view* (GLAV), and it is a mixture of the two; it combines the GAV and LAV approaches in such a way that queries over the sources are put into correspondence with queries over the global schema.

6.2.1 Query Processing

Irrespective whether the mapping is GAV or LAV (or GLAV), query processing in data integration requires a reformulation step: the query posed over the global schema has to be reformulated in terms of a set of queries over the sources. Nevertheless, the actual realization of query processing in data integration systems is strictly dependent on the method used for the specification of the mapping.

Query processing in GAV can be based on a simple *unfolding* strategy: given a query `q` over the alphabet of the global schema A_G, every element of A_G is substituted with the corresponding query over the sources, and the resulting query is then evaluated on data stored by local sources. Query processing in GAV is reduced to unfolding (and is therefore not complex), if there are no integrity constraints on the global schema. Conversely, if integrity constraints are present, data retrieved from the sources may or may not satisfy such constraints. If constraints are violated, the parts of data that do not violate the constraints may still be of interest, and the query answering process should allow their return as a result. Therefore, introducing integrity constraints in GAV implies dealing with issues related to query answering in the presence of incomplete information, and to query answering in the presence of inconsistent information [37]. However, typically, query answering in GAV has the advantage of leading to simpler query answering mechanisms.

Conversely, in the LAV approach it is easier to add or remove sources from the system, while generally requiring more sophisticated query answering techniques. Specifically, since in the LAV approach sources are modeled as views over the global schema, the problem of processing a query is called *view-based query processing.* There are two approaches to view-based query processing: view-based query rewriting and view-based query answering.

View-based query rewriting consists of reformulating the query into a possibly equivalent expression that refers only to the source structures. Once the rewriting of the query has been computed, it can be directly evaluated over the sources to obtain the answer to the query.

View-based query answering is more direct: besides the query and the mapping definitions, we are also given the extensions of the views over the global schema. The goal is to compute the set of tuples that is the answer set of the query in all databases consistent with the information on the views.

More details on query processing and on the definition of a formal framework for data integration are described in Section 6.5.1. In the following, we provide an example to show how the mapping can be specified and used for query processing. Let us consider a global schema consisting of the following relations:

- `Book(Title, Year, Author)`, representing books with their titles, their years of publication and their authors.
- `Award(Title, Prize)`, representing titles of and prizes won by the books.
- `NonProfessional(Author)`, storing names of authors whose main profession is not writing books.

Let us suppose there are two sources: S_1`(Title, Year, Author)` storing information on books since 1930 by non-professional authors, and S_2`(Title,` `Prize`), storing information on awards won by books since 1970. A global query could ask for "title and prize of books published after 1980", corresponding to the Datalog formulation (see [190]):

$$\texttt{Book(T; 1980; A)} \wedge \texttt{Award(T; P)},$$

where, the query is expressed as the conjunction of two atomic formulas with arguments that are variables (`T, A, P`) and constants (`1980`). A GAV mapping would define the global concepts in terms of the sources by means of the following rules:

- $\texttt{Book(T; Y; A)} \leftarrow S_1\texttt{(T; Y; A)}$
- $\texttt{NonProfessional(A)} \leftarrow S_1\texttt{(T; Y; A)}$
- $\texttt{Award(T; P)} \leftarrow S_2\texttt{(T; P)}$

The global query $\texttt{Book(T; 1980; A)} \wedge \texttt{Award(T; P)}$ is processed by means of unfolding, i.e., by expanding the atoms according to their definitions until we come up with source relations. Therefore, in this case, the unfolding process leads to the following query, expressed in terms of source schemas:

$$S_1\texttt{(T; 1980; A)} \wedge S_2\texttt{(T; P)}.$$

Conversely, in the case of an LAV mapping, rules define the concepts in the local source schemas in terms of the global schema as follows:

- $S_1\texttt{(T; Y; A)} \leftarrow \texttt{Book(T; Y; A)} \wedge \texttt{NonProfessional(A)} \wedge \texttt{Y} \geq \texttt{1930}$
- $S_2\texttt{(T; P)} \leftarrow \texttt{Book(T; Y; A)} \wedge \texttt{Award(T, P)} \wedge \texttt{Y} \geq \texttt{1970}$

The query on the global schema is processed by means of an inference mechanism aiming to reexpress the atoms of the global view in terms of atoms at the sources. Therefore, in this case, the inference process leads to the following query, expressed in terms of source schemas:

$$S_1\texttt{(T; 1980; A)} \wedge S_2\texttt{(T; P)}$$

This is the same query derived as a result of the unfolding process; but an inference procedure has been used instead.

6.3 Techniques for Quality-Driven Query Processing

In this section we provide an overview of several proposals to perform quality-driven query processing, which returns an answer to a global query, by explicitly taking into account the quality of data provided by local sources; however, several other techniques are present in the literature, e.g., [25, 24].

6.3.1 The QP-alg: Quality-Driven Query Planning

In this section, we describe the approach presented in [142], which we will refer to as QP-alg in the following. The mapping between local sources and the global schema is specified by means of *query correspondence assertions* (QCAs) that have the general form

$$\texttt{MQ} \leftarrow \texttt{Si.vj} \leftarrow \texttt{WQ},$$

where (i) `MQ` is the mediator query and is a conjunctive query, (ii) `Si.vj` denotes an arbitrary view `vj` on the source `Si`, and (iii) `WQ` is the wrapper query. The mapping can be classified GLAV, as a query on the global schema is defined in terms of a query on the sources.

Three classes of data quality dimensions, called *information quality criteria* (IQ criteria), are defined:

- *Source-specific criteria*, defining the quality of a whole source. Examples of such criteria are *reputation* of the source, based on users' personal preferences, and *timeliness*, measured by the source update frequency.
- *QCA-specific criteria*, defining the quality of specific query correspondence assertions. An example of such criteria is *price*, i.e., the price to be paid for the query.
- *User-query specific criteria*, measuring the quality of the source with respect to the answer provided to a specific user query. An example of such criteria is *completeness*, based on the fullness of source relations.

Some IQ criteria metrics are predetermined, others are dynamically calculated, and the result is a set of IQ criteria vectors to be used to rank sources and plans. Note that, in a DBMS, given a query, query plans are constructed that are equivalent in terms of the query result provided; they are then ranked and selected on the basis of a cost model. Conversely, the plans built according to the QP-alg's approach produce different query results, though they are checked to be semantically correct. The phases of QP-alg are shown in Figure 6.2.

The first phase consists of a pruning of the source space by filtering out low quality sources on the basis of source-specific criteria. In order to classify sources on the basis of IQ criteria vectors, a multiattribute decision making method is used, namely, the data envelopment analysis [44].

The second phase creates plans by exploiting the fact that QCAs are actually views over the mediator schema, and thus basic data integration results for query answering using views can be exploited [117].

The third phase first evaluates the quality of QCAs (step 1 in plan selection in Figure 6.2). Specifically, QCA-specific criteria and user-query specific criteria are calculated for each QCA. Then, the quality of a plan is evaluated (step 2 in plan selection in Figure 6.2) by relying on a procedure similar to cost models for DBMSs. A tree is built for each plan, with QCAs as leaves and join operators as inner nodes. The IQ vector is recursively calculated for a node, starting from its children nodes. A set of "merge" functions for each quality criterion is defined in order to combine IQ vectors. As an example, the merge function for the price criterion is defined as the sum of both the right child and the left child of a given node, meaning that both queries must be made. In Figure 6.3, an example is shown, detailing how the price of a plan P_i is computed.

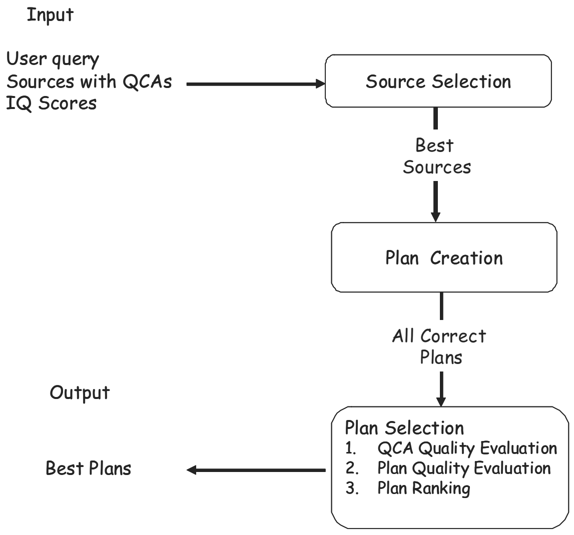


Fig. 6.2. Phases of the QP-alg approach

Then, plan ranking is performed by means of the simple additive weighting (SAW) method (step 3 in plan selection in Figure 6.2). Specifically, the final IQ score for a plan is computed as the weighted sum of scaled criteria, where weights represent the "importance" of each criterion to the user. Finally, the best plans, according to the performed ranking, are returned.

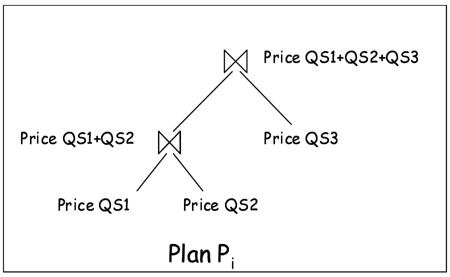


Fig. 6.3. Example of price computation for the plan P_i

6.3.2 DaQuinCIS Query Processing

The DaQuinCIS system, described in [175], is a framework for dealing with data quality in cooperative information systems. A module of the system, the *data quality broker*, is a data integration system. While the overall DaQuinCIS system and its modules will be described in detail in Chapter 8, in this section we focus on the proposed query answering process, which is one of the functionalities of the Data Quality Broker.

The main idea of the DaQuinCIS approach is to make cooperating organizations export not only data that they intend to exchange with other organizations, but also metadata, that characterize their quality level. To this extent, a specific semistructured data model is proposed, called D^2Q. The model is extensively described in Chapter 3. On the basis of such quality characterization of exported data, user queries are processed so that the "best quality" answer is returned as a result.

Queries on the global schema are processed according to the GAV approach by unfolding, i.e., by replacing each atom of the original query with the corresponding view on local data sources. When defining the mapping between concepts of the global schema and concepts of the local schemas, while the extension of global-level concepts can be retrieved by multiple sources, the mapping is actually defined to retrieve the union of local source extensions. Such a mapping definition stems directly from the assumption that the same concept can have different extensions at a local source level due to data quality errors. Therefore, when retrieving data, they can be compared and a best quality copy can be either selected or constructed.

More specifically, query processing in DaQuinCIS is performed by the following sequence of steps:

1. *Query unfolding.* A global query $\mathcal{Q}$ is unfolded according to a static mapping that defines each concept of the global schema in terms of the local sources; this mapping is defined in order to retrieve all copies of the same data that are available, i.e., exported by the cooperating organizations according to the D^2Q model. Therefore, the query $\mathcal{Q}$ is decomposed into $\mathcal{Q}_1, \ldots, \mathcal{Q}_k$ queries to be posed over local sources. Such queries are then executed to return a set of results $\mathcal{R}_1, \ldots, \mathcal{R}_k$ (see Figure 6.4).
2. *Extensional checking.* In this step, a record matching algorithm is run on the set $\mathcal{R}_1 \cup \mathcal{R}_2 \cup \ldots \cup \mathcal{R}_k$. The result of the running of the record matching algorithm is the construction of a set of clusters composed by records referring to the same real-world objects $\mathcal{C}_1, \ldots, \mathcal{C}_z$ (see Figure 6.4, middle).
3. *Result building.* The result to be returned is built by relying on a *best quality default semantics*. For each cluster, a best quality representative is either selected or constructed. Each record in the cluster is composed of couples in which a quality value `q` is associated with each field value `f`. The best quality record for each cluster is selected as the record having the best quality values in all fields, if such a record exists. Otherwise, a best

quality record is constructed by composing the fields that have the highest quality from records within the same cluster. Once representatives for each cluster have been selected, the result $\mathcal{R}$ is constructed as the union of all cluster representatives (see Figure 6.4, right). Each quality value q is a vector of quality values corresponding to the different quality dimensions. For instance, q can include values for accuracy, completeness, consistency, and currency. These dimensions have potentially different scales; therefore, a scaling step is needed. Once scaled, those vectors need to be ranked. Therefore a ranking method must also be applied. Both scaling and ranking problems have well-known solutions, e.g., multi-attribute decision making methods, like AHP [170].

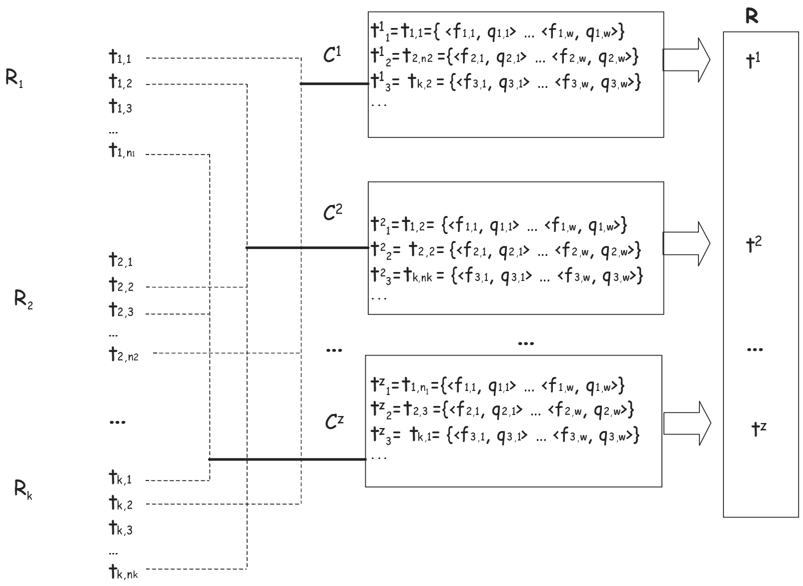

R_1
$t_{1,1}$
$t_{1,2}$
$t_{1,3}$
...
$t_{1,n1}$
R_2
$t_{2,1}$
$t_{2,2}$
$t_{2,3}$
...
$t_{2,n2}$
...
R_k
$t_{k,1}$
$t_{k,2}$
$t_{k,3}$
...
$t_{k,nk}$
C^1
C^2
...
C^z
$t^1_1=t_{1,1}=\{<f_{1,1}, q_{1,1}> \ldots <f_{1,w}, q_{1,w}>\}$
$t^1_2=t_{2,n2}=\{<f_{2,1}, q_{2,1}> \ldots <f_{2,w}, q_{2,w}>\}$
$t^1_3=t_{k,2}=\{<f_{3,1}, q_{3,1}> \ldots <f_{3,w}, q_{3,w}>\}$
...
$t^2_1=t_{1,2}=\{<f_{1,1}, q_{1,1}> \ldots <f_{1,w}, q_{1,w}>\}$
$t^2_2=t_{2,2}=\{<f_{2,1}, q_{2,1}> \ldots <f_{2,w}, q_{2,w}>\}$
$t^2_3=t_{k,nk}=\{<f_{3,1}, q_{3,1}> \ldots <f_{3,w}, q_{3,w}>\}$
...
...
$t^z_1=t_{1,n_1}=\{<f_{1,1}, q_{1,1}> \ldots <f_{1,w}, q_{1,w}>\}$
$t^z_2=t_{2,3}=\{<f_{2,1}, q_{2,1}> \ldots <f_{2,w}, q_{2,w}>\}$
$t^z_3=t_{k,1}=\{<f_{3,1}, q_{3,1}> \ldots <f_{3,w}, q_{3,w}>\}$
...
R
t^1
t^2
...
t^z

Fig. 6.4. The query result construction in DaQuinCIS

6.3.3 Fusionplex Query Processing

Fusionplex[135] models a data integration system by (i) a relational global schema $\mathtt{D}$; (ii) a set of relational local sources (D_i, d_i), where d_i is the instance of the local schema D_i; and (iii) a set of schema mappings (D, D_i).

The mapping definition is GLAV, i.e., views on the global schema are put in correspondence with views on schema of the local sources. In FusionPlex, it is assumed that the *schema consistency assumption* holds, meaning that there are no modeling errors at the local sources, but only modeling differences. Instead, it is assumed that the *instance inconsistency assumption* holds, meaning that the same instance of the real world can be represented differently in the various local sources due to errors. In order to deal with such instance-level inconsistencies, Fusionplex introduces a set of metadata, called *features*, about the sources to be integrated. As better detailed in Section 6.4.2, source features include time stamp, availability, and accuracy. The data integration framework definition presented above is extended by including features into the definition of schema mappings. Specifically, the mappings are triples consisting of a global schema view $\mathtt{D}$, a local schema view $\mathtt{D}_i$, and, in addition the features associated with the local view. Fusionplex includes an extension of the relational algebra that takes into account the association of a set of features $F = \{F_1 \ldots F_n\}$ with source relations. For instance, the extended cartesian product concatenates the database values of the participating relations, but fuses their feature values. The fusion method depends on the particular feature. Therefore, the availability value of the new tuple is the product of the availability values of the input tuples; the time stamp is the minimum of the input time stamps; and so on. In this setting, query processing is performed in several steps:

1. Given a query Q, the set of *contributing views* is identified. First, the sets of attributes of the query and each contributing view are intersected. If the intersection is empty, the contributing view is not relevant. Next, the selection predicates of the query and the contributing view are joined. If the resulting predicate is true, then the contributing view is considered relevant to the query.
2. Once relevant contributing views are identified, *query fragments* are derived as the unit of information suitable for populating the answer to the query. A query fragment results from the removal from the contributing view of all tuples and attributes that are not requested in the query, and from the addition of null values for the query attributes that are missing from the contributing view. As an example, in Figure 6.5, two contributing views, $\mathtt{C}_1$ and $\mathtt{C}_2$, are shown, and the corresponding query fragments, $\mathtt{QF}_1$ and $\mathtt{QF}_2$.
3. From each relevant contributing view, a single query fragment is constructed, where some of these fragments may be empty. The union of all nonempty query fragments is termed a *polyinstance* of the query. Intuitively, a polyinstance includes all the information derived from the data sources in response to a user query.

In order to provide a unique answer to the query Q for the user, instance-level conflicts present in the polyinstance must be solved. Once polyinstances

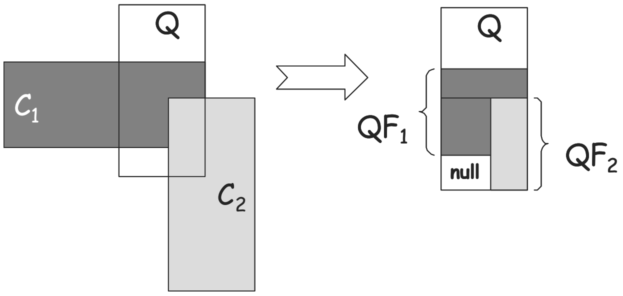


Fig. 6.5. Example of query fragment construction from contributing views

have been constructed a strategy for conflict detection and resolution is applied, as described in Section 6.4.2.

6.3.4 Comparison of Quality-Driven Query Processing Techniques

In Figure 6.6, a comparison of the query processing techniques described is shown. The techniques are compared according to the following features:

- *Quality metadata*, showing that each technique is based on a set of metadata that support the query processing activity.
- *Granularity of the quality model* that represents data elements quality metadata can be associated with. QP-alg associates quality metadata not only with sources but also with query correspondence assertions and user queries. DaQuinCIS exploits the flexibility of a semistructured data model for quality association at various granularity levels. Fusionplex allows association only at a source level.
- *Type of mapping*, showing that both QP-alg and Fusionplex have a GLAV approach to the mapping definition, while DaQuinCIS has a GAV approach.
- *Support to quality algebra*, meaning that quality values associated with local source data need to be "combined" by means of specific algebraic operators. As described in Chapter 4, Section 4.2, there are some research proposals in this direction, but it is still an open problem. Some attempts toward the algebraic manipulation of quality values are present in the merge functions of QP-alg and in the extension of the relational operators of Fusionplex.

6.4 Instance-level Conflict Resolution

Instance-level conflict resolution is a major activity in data integration systems. No data integration system can return answers to user queries if these

Techniques	Quality Metadata	Granularity of Quality Characterization	Type of Mapping	Quality Algebra Support
QP-alg	YES	Source, Query Correspondences Assertions, User Queries	GLAV	Preliminary
DaQuinCIS Query Processing	YES	Each data element of a semistructured data model	GAV	No
FusionPlex Query Processing	YES	Source	GLAV	Preliminary

Fig. 6.6. Comparison of quality-driven query processing techniques

types of conflicts are not solved. As data integration typically deals with heterogeneous and autonomous sources, instance-level conflicts are very common and frequent. Unfortunately, most of the existing data integration solutions have simplifying assumptions regarding conflicts on data values.

In this section, after a classification of these conflicts (Section 6.4.1), we describe some of the existing proposals of instance-level conflict resolution techniques (Section 6.4.2), and we conclude with a comparison between techniques (Section 6.4.3).

6.4.1 Classification of Instance-Level Conflicts

As already mentioned in Section 6.1, in order to integrate data coming from distinct data sources, problems caused by technological, schema, and instance-level heterogeneities need to solved. In the following section, we briefly describe conflict originating from schema heterogeneities, called *schema-level conflicts*, while the latter part is devoted to the description of conflicts due to instance-level heterogeneities, called *instance-level conflicts*.

Schema-level conflicts have been extensively studied (see [109]), and include

- *heterogeneity conflicts*, occurring when different data models are used;
- *semantic conflicts*, regarding the relationship between model element extensions. For instance, a `Person` entity may have different extensions in different sources that may be disjoint, partially overlapping, including one into another, or completely overlapping;
- *description conflicts*, concerning the description of concepts with different attributes. These conflicts include different formats, different attribute types, and different scaling. These conflicts are on the boundary between schema level conflicts and instance-level conflicts; for instance, in [75], such conflicts are classified as data value conflicts. We prefer to consider description conflicts at a schema-level because they are actually caused by different design choices of data schemas, though such choices certainly have an impact on values to be integrated;

- *structural conflicts*, regarding different design choices within the same model. For instance, such conflicts may occur if one source represents an `Address` as an entity and another source represents it as an attribute.

In contrast with schema-level conflicts, instance-level conflicts have received much less attention, and only recently has the importance of these types of conflicts increased, due to the primary role they play in the data integration processes. Instance-level conflicts are due to poor quality of data; they occur because of errors in the data collection process or the data entry process or because sources are not updated.

According to the granularity of the model element, instance-level conflicts can be distinguished into *attribute conflicts* and *key conflicts*, also called *entity* or *tuple conflicts*. Some works, e.g., [118], also consider *relationship conflicts* that are particularly meaningful at a conceptual level. In the following, we will focus on attribute and key conflicts, as they are the principal conflict types involved in data integration processes.

Let us consider two relational tables, $\mathtt{S}_1(\mathtt{A}_1, \ldots, \mathtt{A}_k, \mathtt{A}_{k+1}, \ldots, \mathtt{A}_n)$ and $\mathtt{S}_2(\mathtt{B}_1, \ldots, \mathtt{B}_k, \mathtt{B}_{k+1}, \ldots, \mathtt{B}_m)$, where $\mathtt{A}_1 = \mathtt{B}_1 \ldots \mathtt{A}_k = \mathtt{B}_k$. Let the same real world entity be represented by the tuple $\mathtt{t}_1$ in $\mathtt{S}_1$ and by the tuple $\mathtt{t}_2$ in $\mathtt{S}_2$, and let $\mathtt{A}_i = \mathtt{B}_i$; the following conflicts can be defined:

- An *attribute conflict* occurs iff

$$\mathtt{t}_1.\mathtt{A}_i \neq \mathtt{t}_2.\mathtt{B}_i.$$

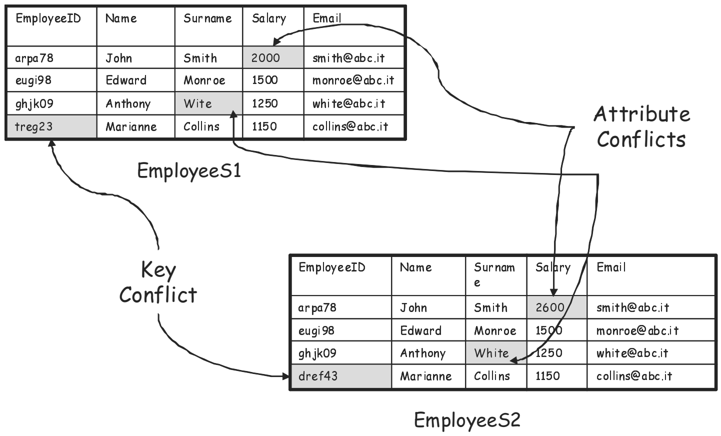
- Let us further suppose that $\mathtt{A}_i$ is a primary key for $\mathtt{S}_1$ and $\mathtt{B}_i$ is a primary key for $\mathtt{S}_2$. A *key conflict* occurs iff

$$\mathtt{t}_1.\mathtt{A}_i \neq \mathtt{t}_2.\mathtt{B}_i \text{ and } \mathtt{t}_1.\mathtt{A}_j = \mathtt{t}_2.\mathtt{B}_j,$$

  for all $\mathtt{j}$ ranging from 1 to $\mathtt{k}$, and $\mathtt{i} \neq \mathtt{j}$.

In Figure 6.7, several examples of both attribute and key conflicts are shown. In the figure, two relations, `EmployeeS1` and `EmployeeS2`, represent information about employees of a company. Notice that we assume there is no schema-level conflict, i.e., the two relations have exactly the same attributes and the same extension. Nevertheless, they present instance-level conflicts. Two attribute value conflicts are shown, concerning the `Salary` of the employee `arpa78` and the `Surname` of the employee `ghjk09` in the two relations. A key-level conflict is also shown between the employee `Marianne Collins`, as identified in the relation `EmployeeS1` and as identified in relation `EmployeeS2`, assuming that the two tuples represent the same real-world object.

Instance-level conflicts can be present in both virtual and materialized integration. In virtual data integration, a theoretical formulation of the problem has been proposed. Specifically, the cited key and attribute conflicts have been formally specified as a violation of integrity constraints expressed over the



EmployeeID	Name	Surname	Salary	Email
arpa78	John	Smith	2000	smith@abc.it
eugi98	Edward	Monroe	1500	monroe@abc.it
ghjk09	Anthony	Wite	1250	white@abc.it
treg23	Marianne	Collins	1150	collins@abc.it

EmployeeS1

EmployeeID	Name	Surname	Salary	Email
arpa78	John	Smith	2600	smith@abc.it
eugi98	Edward	Monroe	1500	monroe@abc.it
ghjk09	Anthony	White	1250	white@abc.it
dref43	Marianne	Collins	1150	collins@abc.it

EmployeeS2

Fig. 6.7. An example of key- and attribute-level conflicts

global schema representing the integrated view. More details on the theoretical perspective on inconsistencies in data integration are provided in Section 6.5.2.

In the next section, we will describe several techniques proposed in order to solve instance-level conflicts.

6.4.2 Overview of Techniques

Techniques that deal with instance-level conflicts can be applied in two different phases of the life cycle of a data integration system, namely, at *design time* and at *query time*. In both cases, the actual conflicts occur at query time; however, the design time approaches decide the strategy to follow for fixing conflicts before queries are processed, i.e., at the design stage of the data integration system. The techniques operating at query time incorporate the specification of the strategy to follow within query formulation.

A proposal for solving conflicts at design time can be found in [55]. The main idea is to resolve attribute conflicts by means of aggregation functions to be specified for each attribute that may involve conflicts during query execution time.

Design time techniques have a major optimization problem, as outlined in [218]. Let us consider the example shown in Figure 6.7, and suppose that it is specified at design time, for the `Salary` attribute, that in the case of conflicts the minimum salary must be chosen. Given a global schema, `Employee(EmployeeID, Name, Surname, Salary, Email)`, let us consider the following query:

```
SELECT EmployeeID, Email
FROM   Employee
```

```
WHERE  Salary < 2000
```

Since the `Salary` attribute is involved in the query, all employees must be retrieved in order to compute the minimum salary, not only employees with `Salary < 2000`, even if no conflicts on salary occur. Therefore, conflict resolution at design time may be very inefficient.

Query time conflict resolution techniques have been proposed to overcome such performance inefficiencies. Furthermore, query time techniques are characterized by greater flexibility, since, as we will see, they allow those who formulate the query to indicate a specific strategy to adopt for conflict resolution. Given a user query posed on the global schema, query time techniques deal with key and/or attribute conflicts that may occur on the data retrieved as results.

Key conflicts require the application of object identification techniques, described in detail in Chapter 5. With reference to the example shown in Figure 6.7, object identification techniques will match the tuple `treg23` from `EmployeeS1` with the tuple `dref43` from `EmployeeS2` by comparing the attribute values of the two tuples in order to determine whether the "Marianne Collins" represented in the two sources is the same person. After a positive matching decision, the tuples referring to "Marianne Collins" will be considered a single tuple, and a unique key will be chosen to identify the tuple, thereby solving the key conflict. If the matching decision is negative, no key conflict occurs.

With respect to attribute conflicts, several techniques for solving them have been proposed:

- SQL-based conflict resolution [141];
- Aurora [218];
- Fusionplex [135];
- DaQuinCIS [174];
- FraSQL-based conflict resolution [176]; and
- OO_{RA} [118].

In the following we describe the details of such techniques; however, several other proposals are present in the literature, including [81, 154]. Before providing the detailed description, we illustrate which are the "abstract" steps to be followed for solving attribute-level conflicts.

Let us consider again the example in Figure 6.7, and let us suppose the following query is formulated over the global schema `Employee(EmployeeID, Name, Surname, Salary, Email)`:

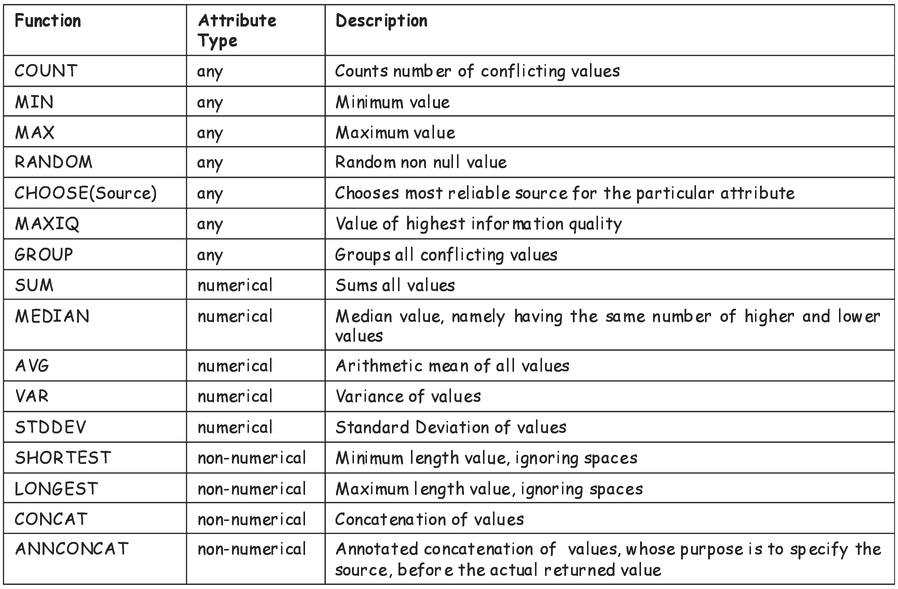
```
SELECT Salary
FROM   Employee
WHERE  Name = "John" AND Surname = "Smith"
```

In order to return a result to this type of query, the attribute conflict between the two values for John Smith's salary stored in the relations `EmployeeS1` and `EmployeeS2` must be solved.

A solution to this problem is to *declaratively* specify how to deal with such conflicts. A declarative specification consists of

- a set of conflict resolution functions that, on the basis of the specific attributes involved in the conflict, can select the most appropriate value;
- a set of strategies to deal with conflicts, corresponding to different tolerance degrees; and
- a query model that can take into account possible conflicts directly, i.e., with specific extensions, such as adhoc functions dealing with conflicts, or indirectly, i.e., without specific extensions.

A *resolution function* takes two (or more) conflicting values of an attribute as input and outputs a value that must be returned as the result to the posed query. Common resolution functions are `MIN` and `MAX`. To these, resolution functions that are specific to some attribute types can be added. For instance, for numerical attribute types, `SUM` and `AVG` can be used. For non numerical attributes, further resolution functions can be identified, such as `CONCAT`. In [141], a resolution function `MAXIQ` is proposed. Assuming the presence of a data quality model that associates quality values to model elements (e.g., attributes), the resolution function `MAXIQ` returns the value with the highest quality. In Figure 6.8, conflict resolution functions are summarized, as proposed in [141]. Some functions are the usual aggregation functions; others serve the specific purpose of resolving conflicts.



Function	Attribute Type	Description
COUNT	any	Counts number of conflicting values
MIN	any	Minimum value
MAX	any	Maximum value
RANDOM	any	Random non null value
CHOOSE(Source)	any	Chooses most reliable source for the particular attribute
MAXIQ	any	Value of highest information quality
GROUP	any	Groups all conflicting values
SUM	numerical	Sums all values
MEDIAN	numerical	Median value, namely having the same number of higher and lower values
AVG	numerical	Arithmetic mean of all values
VAR	numerical	Variance of values
STDDEV	numerical	Standard Deviation of values
SHORTEST	non-numerical	Minimum length value, ignoring spaces
LONGEST	non-numerical	Maximum length value, ignoring spaces
CONCAT	non-numerical	Concatenation of values
ANNCONCAT	non-numerical	Annotated concatenation of values, whose purpose is to specify the source, before the actual returned value

Fig. 6.8. Resolution functions as proposed in [141]

The *tolerance strategies* allow the user to define the degree of conflict permitted. For example, it is possible to specify that on a specific attribute no conflicts are admitted. This means that all values returned by the sources on that attribute must be aligned. As another example, it may be possible to specify that in the case of conflicts, a randomly chosen value among the conflicting ones be proposed as the result. As another tolerance strategy, a threshold value may be specified for distinguishing tolerable conflicts from intolerable ones. For instance, a conflict on two values for the `Name` attribute such as `Michael` and `Maichael`, that have a reciprocal edit distance of one character, can be tolerated since it is very easy to transform `Maichael` into `Michael`, by deleting simply one character. In contrast, for a numerical attribute like `Salary`, even a one digit distance may be intolerable.

With respect to the *query model*, it is possible to appropriately use SQL to specify how to solve conflicts [141], or to use adhoc extensions such as the ones proposed in [118] and [218].

The next sections will describe several techniques for conflict resolution that instantiate the abstract steps presented.

SQL-Based Conflict Resolution

The approach proposes formulating queries in SQL, exploiting the capabilities of current database systems. Three possible strategies are discussed, based on three SQL operations:

- *Group*, where by using the `Group by` SQL statement, a query is specified that groups tuples on the basis of one or more group attributes. Then, an aggregated function is specified to select conflicting values appropriately. For instance,

  ```
  SELECT   EmployeeId, min(Salary)
  FROM     Employee
  GROUP BY EmployeeId
  ```

  The main disadvantage of this approach is that only the aggregation functions supported by SQL can be used.
- *Join*, which considers the union of two sources, and partitions it into three sets: the intersection of the two sources, the tuples only in the first source, and the tuples only in the second source. Then, the merging query is expressed on each of these parts, and, finally, the results are merged. The first query is expressed on the intersection:

  ```
  SELECT EmployeeID, min(Employee1.Salary, Employee2.Salary)
  FROM   Employee1, Employee2
  WHERE  Employee1.EmployeeId = Employee2.EmployeeId
  ```

  This query has the advantage that the resolution is no longer an aggregate function, but a scalar one. This extends the possibility of using user-defined functions, thereby enlarging the spectrum of possible resolution functions while continuing to be compliant with most database systems that allow user-defined scalar functions. The following query selects the tuples of the first source that are not in the second:

```
SELECT EmployeeId, Price
FROM   Employee1
WHERE  Employee1.EmployeeId NOT IN
       (SELECT EmployeeID
        FROM Employee2)
```

The query to select the tuples of the second source that are not in the first source is similar to the above one. The query to merge is simply the combination of the results of all queries through the `UNION` operator. The main disadvantage of this approach is the complexity of the queries, because the number of partitions increases exponentially with the number of sources. The length and complexity of queries may become prohibitive.

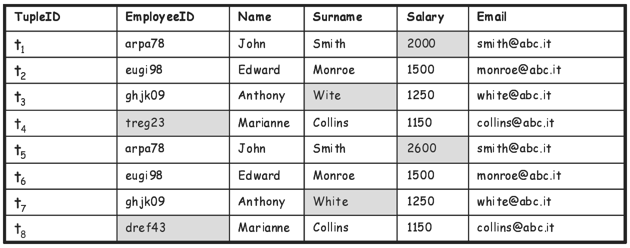
- *Nested Join*, an improvement over the previous method, which can be performed when resolution functions are associative. Given N sources to be merged, the idea is to first merge two, then merge this with a third, and so on. With this approach, queries grow linearly, but still remain complex.

Aurora

Aurora is a mediation-based data integration system. The approach proposes a conflict-tolerant query model for conflict resolution at a desired degree. The conflict-tolerant query model has the following features:

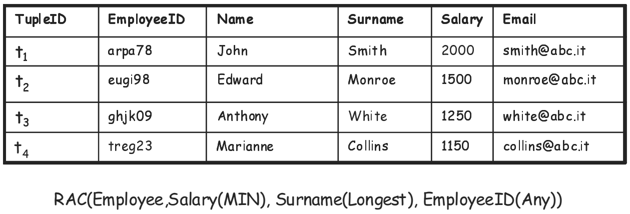
- Two operators, for attribute conflict resolution, called *resolve attribute-level conflict* (RAC) and for tuple conflict resolution, called *resolve tuple-level conflict*(RTC). The operators take a resolution function as parameter. For example, consider the global population of the relation `Employee`, shown in Figure 6.9, which represents the global instance resulting from the integration of the two relations `EmployeeS1` and `EmployeeS2` shown in Figure 6.7. An example of how the operator RAC works is reported in Figure 6.10, where the specified resolution functions are `MIN` for `Salary`, `LONGEST` for `Surname`, and `ANY` for `EmployeeID`. An example for the RTC operator is shown in Figure 6.11, where the resolution function is `ANY`, and the tuple conflicts are solved choosing tuple `dref43`.
- Three strategies for conflict resolution, namely, `HighConfidence`, `RandomEvidence`, and `PossibleAtAll`. These strategies allow the user to define the degree of conflicts permitted, and are used in conjunction with the previously described operators when formulating queries. `HighConfidence` allows us to specify that no conflicts on a specific attribute are admitted. This means that all values returned by the sources on that attribute must be aligned. `RandomEvidence` specifies that in the

case of conflicts, a runtime function has to select a value to be returned. `PossibleAtAll` returns all values that correctly answer the query, independently of conflicts.



TupleID	EmployeeID	Name	Surname	Salary	Email
t_1	arpa78	John	Smith	2000	smith@abc.it
t_2	eugi98	Edward	Monroe	1500	monroe@abc.it
t_3	ghjk09	Anthony	Wite	1250	white@abc.it
t_4	treg23	Marianne	Collins	1150	collins@abc.it
t_5	arpa78	John	Smith	2600	smith@abc.it
t_6	eugi98	Edward	Monroe	1500	monroe@abc.it
t_7	ghjk09	Anthony	White	1250	white@abc.it
t_8	dref43	Marianne	Collins	1150	collins@abc.it

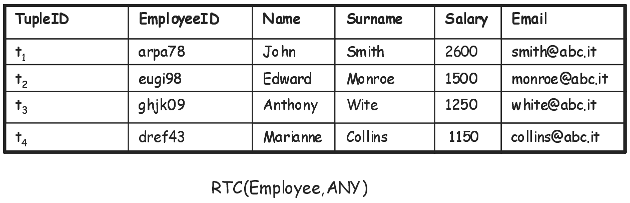
Fig. 6.9. Instance of the global relation `Employee`



TupleID	EmployeeID	Name	Surname	Salary	Email
t_1	arpa78	John	Smith	2000	smith@abc.it
t_2	eugi98	Edward	Monroe	1500	monroe@abc.it
t_3	ghjk09	Anthony	White	1250	white@abc.it
t_4	treg23	Marianne	Collins	1150	collins@abc.it

RAC(Employee,Salary(MIN), Surname(Longest), EmployeeID(Any))

Fig. 6.10. Resolution of attribute conflicts



TupleID	EmployeeID	Name	Surname	Salary	Email
t_1	arpa78	John	Smith	2600	smith@abc.it
t_2	eugi98	Edward	Monroe	1500	monroe@abc.it
t_3	ghjk09	Anthony	Wite	1250	white@abc.it
t_4	dref43	Marianne	Collins	1150	collins@abc.it

RTC(Employee, ANY)

Fig. 6.11. Resolution of tuple conflicts

The conflict-tolerant query model is built on tuple-level conflicts only, but the user is allowed to specify attribute-level conflict resolution. Some examples of conflict-tolerant queries are as follows:

-
  ```
  Q1: SELECT EmployeeID, Name (ANY), Salary[MIN]
      FROM   Employee
      WHERE  Salary>1800
      WITH   HighConfidence
  ```

- Q2:
```
SELECT [ANY]EmployeeID, Name, Salary
FROM   Employee
WHERE  Salary>1800
WITH   RandomEvidence
```

Both queries select employees with `Salary` greater than 1800 euros. If there is a conflict, `Q1` selects employees whose `Salary` value is greater than 1800 in all sources. Therefore, based on Figure 6.9, the tuples $\mathtt{t}_1$ and $\mathtt{t}_5$ are selected. Then, applying the resolution function `MIN` on `Salary`, the returned tuple will have the `Salary` value of $\mathtt{t}_1$, namely, 2000. `Q2` selects a random `Salary` value, and, if it is greater than 1800, it is returned as a result. Then, the `ANY` tuple resolution function is applied as specified in the selection clause. Based on Figure 6.9, a random value between the `Salary` value of $\mathtt{t}_1$ and $\mathtt{t}_5$ is returned.

Fusionplex and DaQuinCIS

The two approaches to conflict resolution adopted in the Fusionplex and DaQuinCIS systems are similar. They both resolve attribute conflicts on the basis of metadata associated with data of local sources.

Fusionplex proposes the following metadata, called *features* :

- *time stamp*, representing the time the source was validated in the system;
- *cost*, which can be transmission time over the network, or money to be paid for information or both;
- *accuracy*, evaluated according to a probabilistic approach;
- *availability*, probability that the information is randomly available; and
- *clearance*, corresponding to the clearance level needed to access the information.

In Fusionplex, the features are associated with sources as a whole, with the restrictive assumption that data in sources are homogeneous with respect to a specific feature.

DaQuinCIS proposes the following metadata, referred to as *dimensions*:

- *accuracy*, concerning the syntactical accuracy of data values;
- *currency*, considering the degree of up-to-dateness of values;
- *consistency*, measuring intrasource integrity constraints; and
- *completeness*, counting the number of null values.

The D^2Q data model, described in detail in Chapter 3, is semistructured, and permits the association of metadata with data elements of different granularity, and therefore, with single values, as well as with attributes and all other model elements.

An example of the extended SQL statements that can be defined in Fusionplex is

```
SELECT EmployeeID, Salary
FROM   EmployeeS1, EmployeeS2
WHERE  EmployeeS1.EmployeeID=EmployeeS2.EmployeeID
USING  cost>0.6
WITH   timestamp as 0.5
```

Considering an XML-based representation of the two relations `EmployeeS1` and `EmployeeS2`, an example of a DaQuinCIS query, expressed in XQuery [29], is

```
FOR    $i in input()//EmployeeS1
FOR    $j in input()//EmployeeS2
WHERE  ($i/EmployeeID=$j/EmployeeID) and
       quality($i/Salary)>0.7 and quality($j/salary)>0.7
RETURN ($i/Name,$i/Salary)
```

As described, attribute conflict resolution is based on metadata in both Fusionplex and DaQuinCIS. Also, both systems have a step in which, upon issuing a user query, all the significant instances answering the query are collected and grouped into clusters of different copies of the same objects. Then, in both systems, a resolution policy is applied in order to produce selected tuples to be included in the result.

The two systems differ in the process for building the final result. In Fusionplex, as described in Section 6.3.3, the phase in which results are collected from local sources terminates with the construction of a polyinstance, upon which a conflict resolution strategy is applied. Conflict resolution is performed in two phases: in the first phase a utility function is used to take user preferences into account, while in the second phase the actual fusion is performed.

With reference to the first phase, users can specify the importance they assign to each feature. Then, an overall utility function consisting of the weighted sum of the feature values of a source is calculated, and a first pruning of sources is done on the basis of a fixed utility threshold.

With respect to the second phase, resolution of inconsistencies can be done either on the basis of their features, called *feature-based resolution*, or on the basis of the data, called *content-based resolution.*

A resolution policy consists of the sequential selection of

- *elimination functions*, which can be feature-based or selection-based. Examples of elimination functions are `MIN` and `MAX`; `MAX(timestamp)` and `MIN(cost)` are examples of feature-based elimination functions, while `MAX(Salary)` is an example of a content-based elimination function.
- *fusion functions.* Fusion functions are always content-based; examples are `ANY` and `AVERAGE`.

Note that the resolution policy is completely specified by users according to their specific requirements. Moreover, Fusionplex admits three tolerance levels: no resolution, pruning of polytuples, and selective attribute resolution. The no resolution policy aloows an answer with conflicts to be returned to the user. The pruning of polytuple policy removes tuples that either do not satisfy the feature selection predicate or are below the utility threshold. The selective attribute resolution forces resolution on some attributes only.

In the DaQuinCIS system, the reconciled result is produced according to the process described in Section 6.3.2, and it is completely based on quality values associated with data on the basis of the D^2Q model.

FraSQL-Based Conflict Resolution

The approach proposes an extension of a multidatabase query language, called *FraSQL*, which provides operations for transformation and integration of heterogeneous data. The main idea is to use grouping for duplicate elimination and aggregation for conflict resolution. For conflict resolution, FraSQL provides both *user-defined aggregation* and *user-defined grouping*. User defined aggregation is useful for conflict resolution, allowing for the selection of a representative value from a group of values corresponding to the same real-world object. The grouping of values is performed by means of user-defined grouping. User defined grouping can be of two types: (i) *context free* and (ii) *context aware*. Context free grouping is the usual approach, as in SQL standards, with, in addition, the possibility of using external functions. The following query shows the usage of a context free user-defined grouping [176]:

```
SELECT   avg (Temperature),rc
FROM     Weather
GROUP BY regionCode(Longitude,Latitude) AS rc
```

where `regionCode` is an external function that computes the region from its geographical position.

Context-aware grouping is proposed in order to overcome some limitations of the current SQL standardized group by operator. Indeed, SQL standardized group by operator works one tuple at a time, not considering possible relationships between grouping tuples. Therefore, in order to have a more flexible grouping, similarity criteria can be introduced that split or merge the group conveniently. As an example, consider the query:

```
SELECT EmployeeID,Salary
FROM   EmployeeS1
GROUP  maximumDifference(Salary,diff=150)
BY CONTEXT
```

The query considers the relation EmployeeS1 shown in Figure 6.7 and groups the tuples as shown in Figure 6.12, generating three sets corresponding to tuples for which the `Salary` values differ by at most 150.

EmployeeID	Salary
arpa78	2000

EmployeeID	Salary
eugi98	1500

EmployeeID	Salary
ghjk09	1250
treg23	1150

Fig. 6.12. Result of the context-aware query as applied to the table EmployeeS1 of Figure 6.7

OO$_{RA}$

Though in the following we focus only on attribute-level conflicts, the model also considers key conflicts and relationship conflicts (see [118] for more details on these two conflict types). The approach distinguishes two types of attribute conflicts, namely, *tolerable conflicts* which can be automatically solved, and *intolerable conflicts*, which have to be solved with human intervention. The two types of conflicts are separated by means of a threshold. An extended object-oriented data model, called OO$_{RA}$, is proposed to handle attribute-level conflicts. The main features of the model with respect to attribute conflict resolution are

- the possibility of specifying thresholds and resolution functions for attribute-level conflict resolution; and
- the representation of original and resolved attribute values.

With respect to the threshold specification and resolution functions, the following three different combinations are considered for a given attribute: (i) threshold predicate and resolution function both unspecified; (ii) specified threshold predicate and unspecified resolution function; and (iii) threshold predicate and resolution function both specified. In case (i), no conflict is tolerated, so if a conflict arises, the resolved attribute value is null. In case (ii), a conflict can arise and can be acceptable, but if it arises, the returned value is `NULL`. In case (iii), there can be tolerable conflicts, and the returned value is computed by the resolution function.

With respect to conflicting values representation, the OO$_{RA}$ approach for every non-identifier attribute represents a triple: original value, resolved value, and conflict type. Conflict type is `NULL` if there is no conflict, `RESOLVABLE` if there is an intolerable conflict, and `ACCEPTABLE` if there is a tolerable conflict. For example, let us consider the following threshold predicate and resolution function applied to the global relation described in Figure 6.9:

```
DEFINE Salary.threshold@EMPLOYEE(s1,s2) = (abs(s1-s2)<=1000)
DEFINE Salary.resolution@EMPLOYEE(s1,s2) = MIN(s1,s2)
```

In this case, the conflict between `t`$_1$ and `t`$_5$ is tolerable, as the differences between the two values for salary are within the specified threshold. The conflict is solved by choosing the value for salary present in tuple `t`$_1$.

As another example, let us consider the following threshold predicate and resolution function, also applied to the relation in Figure 6.9:

```
DEFINE Surname.threshold@EMPLOYEE(s1,s2) = (editDistance(s1,s2)<=1)
DEFINE Surname.resolution@EMPLOYEE(s1,s2) = LONGEST(s1,s2)
```

Still, the conflict between `t`$_3$ and `t`$_7$ is tolerable, and the value for `Surname` stored by tuple `t`$_7$ is returned as a result. In contrast, supposing that the `Surname` value for `t`$_3$ were `Wie`, and the edit distance between `t`$_3$`.Salary` and `t`$_7$`.Salary` greater than 1, an intolerable conflict would have occurred.

6.4.3 Comparison of Instance-level Conflict Resolution Techniques

In Figure 6.13, the different declarative techniques for the resolution of inconsistencies are compared with respect to permitted tolerance strategies and query models. Reviewing the tolerance strategies column, Aurora, Fusionplex, and OO$_{RA}$ propose a degree of flexibility that can be selected once conflicts occur. We recall that the three degrees of flexibility proposed by Aurora are (i) high confidence, meaning that no conflict is tolerated, (ii) random evidence meaning that in the case of conflicts a runtime function will select the value to be returned, and (iii) possible at all, meaning that all values that correctly answer the query must be returned. Similarly to Aurora, Fusionplex admits three tolerance levels: no resolution, pruning of polytuples, and selective attribute resolution. The no resolution policy corresponds to PossibleAtAll; in both approaches, the answer with conflicts is returned to the user. The pruning of polytuple policy, which removes tuples not satisfying the feature selection predicate or the utility threshold, is a more specific case of the RandomEvidence policy and it shares the threshold concept with OO$_{RA}$. The selective attribute resolution involves leaving some (or all) attributes unresolved; it is a specific case of the no resolution policy with higher granularity.

Reviewing the query model column, we see that the SQL-based conflict resolution can rely on SQL. However, it has inefficiencies due to the fact that resolution functions were not considered for the native SQL. Therefore, computing aggregation and expressing SQL statements for them can become very onerous. Both DaQuinCIS and OO$_{RA}$ deal with models that are different from the relational model, namely, with the XML data model and the object-oriented data model, respectively.

Techniques	Tolerance Strategies	Query Model
SQL-Based Conflict Resolution	NO	SQL
Aurora	High Confidence, RandomEvidence, PossibleAtAll	Ad-hoc Conflict Tolerant Query Model
Fusionplex	No resolution strategy, selective attribute resolution	Extended SQL
DaQuinCIS	NO	Extended XML
FraQL-Based Conflict Resolution	NO	Ad-hoc FraQL
OO_{RA}	Thresholds for tolerable and intolerable conflicts	Ad hoc Object Oriented Extension (OO_{RA})

Fig. 6.13. Conflict resolution techniques

6.5 Inconsistencies in Data Integration: a Theoretical Perspective

In this section, we first provide first several basic definitions that formally specify a data integration system (Section 6.5.1). Then, we discuss an example of what inconsistency means on the basis of such formal specifications, and give some hints on specific semantics that have been defined for dealing with inconsistencies (Section 6.5.2).

6.5.1 A Formal Framework for Data Integration

A *data integration system* (*DIS*) [116] can be formally defined as a triple (G, S, M) where

- G is the global schema, expressed in a language L_G over an alphabet A_G.
- S is the source schema[1], expressed in a language L_S over an alphabet A_S.
- M is the mapping between G and S, constituted by a set of assertions of the forms
$$q_S \rightsquigarrow q_G \text{ and}$$
$$q_G \rightsquigarrow q_S,$$
where q_G and q_S are two queries of the same arity, respectively over the global schema G and the source schema S. Queries q_S are expressed in a query language $L_{M,S}$ over the alphabet A_S, and queries q_G are expressed in a query language $L_{M,G}$ over the alphabet A_G.

[1] The source schema in [116] is a collective name that indicates the *set* of source schemas, as introduced in Section 6.2.

In Section 6.2, we provided some examples on how mapping assertions can be specified.

Given a data integration system I=(G, S, M), a *semantics* to it can be assigned by specifying the information content of the global schema G. Let D be a source database for I, i.e., a (set of) database that conforms to the source schema S and satisfies all constraints in S. On the basis of D, we can define the information content of the global schema G. We call *global database* for I any database for G. A global database B is said to be *legal with respect to* D if:

- B is legal with respect to G, i.e., B satisfies all the constraints of G; and
- B satisfies the mapping M with respect to D.

An important notion to be introduced is that of *certain answers*. Given a source database D for I, the answer $q_{I,D}$ to a query q in I with respect to D is the set of tuples t of objects such that t $\in q_B$ for every global database B that is legal for I with respect to D.

The meaning of the sentence "B satisfies the mapping M with respect to D" depends on how to interpret the assertions.

In the LAV case, where mapping assertions have the form $s \rightsquigarrow q_G$, the following cases have been identified:

- *Sound views.* When a source s is *sound*, its extension provides any subset of the tuples satisfying the corresponding view q_G.
- *Complete views.* When a source s is *complete*, its extension provides any superset of the tuples satisfying the corresponding view.
- *Exact Views.* When a source s is *exact*, its extension is exactly the set of tuples of objects satisfying the corresponding view.

In the GAV case, a similar interpretation of mapping assertions can be given, and hence, sound, complete, and exact views can correspondingly be defined. In the following section, we see the role that sound, complete, and exact views can play when dealing with inconsistent answers.

6.5.2 The Problem of Inconsistency

In a data integration system (DIS), beyond the inconsistency problems that are local to sources, inconsistency may arise due to integrity constraints that are specified on the global schema.

Integrity constraints on the global schema represent a fundamental knowledge, as they actually allow one to capture the semantics of the reality. Sources in a DIS are autonomous and independent; indeed, DISs can be seen as a particular case of cooperative information systems, where the cooperation is actually realized by means of data sharing among the distinct sources (see Chapter 1, Section 1.4). Each source in a DIS locally checks for the satisfaction of its own integrity constraints. As a component of a data integration system, each source has to check further if it violates the integrity constraints specified over the global schema. If this happens, it is necessary to set how to deal with such inconsistencies. More specifically, it is not admissible that the whole DIS not provide any answer to a user query if consistency violation occurs. Instead, specific techniques need to be introduced in order to deal with such inconsistencies. In the following, an example of integrity constraint violation is described, and different problems that arise are introduced.

Let us consider a global schema consisting of two relations, representing movies and the actors who have acted in these movies: `Movie(Title, Director)` and `Actor(Name,Surname,Movie)`. Let us assume that a foreign key constraint exists between the attribute `Movie` of `Actor` and the attribute `Title` of `Movie`. Let us further assume that a GAV mapping is defined.

We first consider the case in which both the relations are defined by exact views on the sources, i.e., all and only all the data retrieved from the sources satisfy the global schema. Let us assume we retrieve the following instances:

```
1  <actor(Audrey, Hepburn, Roman Holidays)>
2  <movie(Roman Holidays, Wyler)>
3  <actor(Russel, Crowe, The Gladiator)>
```

Tuple 3 violates the foreign key constraint; therefore, a query asking for all movies would provide no answer, though tuple 2 could be provided as an answer.

If we do not consider exact views but instead consider sound or complete views, it is possible to provide answers. Recall that a view is sound in a GAV mapping if the provided data are a subset of the data satisfying the global schema. A view is complete if it provides a superset of the data satisfying the global schema.

As a second case, we consider the relation `Actor` defined as a complete view and `Movie` as a sound view. In this case, a query asking for all movies would have tuple 2 as an answer, because it is possible to delete some tuples of actors due to soundness, and it is possible to add a tuple <`movie(The Gladiator,` α`)`>, where α is a placeholder for the director's value.

Also in the case of sound or complete views, there are cases in which no answer can be provided. Indeed, if `Actor` were defined by a complete view and `Movie` by a sound view, the foreign key constraint could not be satisfied.

In order to provide consistent answers, when inconsistent databases are retrieved, it is necessary to introduce different semantics for the data integration systems that take into account the possibility of adding or deleting tuples to reinstate consistency. Some works in the direction of defining a semantics for DISs in presence of inconsistencies have been proposed in the literature. All such works are based on the notion of repair, introduced in the setting of inconsistent databases in [9]. Given an inconsistent database, a repair is a database consistent with the integrity constraint which "minimally" differs from the original database, where minimality depends on the semantic criteria adopted to define an ordering among consistent databases (e.g, based on set-containment [9, 86, 38], or cardinality [119]). Other works (see e.g., [32, 39]) generalized this notion to the context of data integration systems, properly taking into account the role of the mapping. Finally, some works have considered the problem for DISs in the presence of preferences specified on the data sources In [57], there is the proposal of a semantics for taking preference criteria into account when trying to solve inconsistencies between data sources in an LAV setting. *Preference criteria* are actually quality criteria specified on data sources. First, a maximally sound semantics is introduced. Given a data integration system `I = (G, S, M)`, the defined semantics considers those interpretations that satisfy G and satisfy the mapping assertion in `M` *as much as possible* with respect to a source model `D` for `I`. Then, the concept of source preference is added, so that among maximally-sound models, only those that refer to sources that are *best* with respect to quality preferences are selected. In [87], a different semantics is introduced, based on the repair of data stored at sources in the presence of a global inconsistency. This choice is an alternative to the choice of repairing global database instances constructed on the basis of the mapping. The semantics introduced in [87] refers to the GAV mapping.

6.6 Summary

Data integration and data quality are two interrelated concepts. On the one hand, data integration can benefit from data quality. Quality-driven query processing techniques have the purpose of selecting and accessing data of the highest quality, thus deriving the maximum benefits from a context with multiple sources with varying quality of their data assets. In open contexts, such as P2P systems, these techniques are becoming increasingly more important, as discussed in Chapter 9 on open problems.

On the other hand, it is intuitive that most data quality problems become evident when data in one source are compared with similar data stored in a different source. Once they are detected, there is the need for appropriate mechanisms that allow a data integration system to perform the query processing function. These techniques are the conflict resolution techniques, which play the significant role of supporting query processing in virtual data integration systems. Note that the choice of solving conflicts at query time is an alternative to the more expensive choice of cleaning data sources *before* they are actually integrated. This would indeed require a data quality improvement activity performed independently by each source, and hence the complexity and the cost would grow.

In materialized data integration, e.g., in data warehouses, a cleaning activity is performed when populating the global schema. As instances gathered by disparate sources typically present instance-level conflicts, conflict resolution techniques can be also effectively applied for the purpose of producing a consistent materialized global instance.

Methodologies for Data Quality Measurement and Improvement

Measuring and improving data quality in a single organization or in a set of cooperating organizations is a complex task. In previous chapters we discussed relevant activities for improving data quality (Chapter 4) and corresponding techniques (Chapters 4, 5, and 6). Several methodologies have been developed in the last few years that provide a rationale for the optimal choice of such activities and techniques. In this chapter we discuss methodologies for data quality (DQ) measurement and improvement from multiple perspectives. Section 7.1 provides basic material, in terms of typical inputs and outputs of methodologies, classifications, finally focusing on the comparison of data and process-driven strategies adopted in methodologies. Section 7.2 deals with assessment methodologies, while Section 7.3 first focuses on the definition of common methodological phases, then describes and compares in terms of the common phases three of the most relevant general purpose methodologies. In Section 7.4 we propose CDQM, an original methodology that at the same time is complete, flexible, and simple to apply; in Section 7.5 CDQM is applied to a case study.

7.1 Basics on Data Quality Methodologies

We define a DQ methodology as a set of guidelines and techniques that, starting from the input information concerning a given reality of interest, defines a rational process for using the information to measure and improve the quality of data of an organization through given phases and decision points. In the rest of the section we focus on inputs and outputs, classifications, and typical strategies adopted in DQ methodologies.

7.1.1 Inputs and Outputs

The different types of input knowledge to a DQ methodology in the most general case are shown in Figure 7.1, where arrows represent generalization hierarchies among concepts; e.g., *Collections of data* can be *Internal groups* or *External sources*, and Internal groups can be *Data flows* or *Databases.*

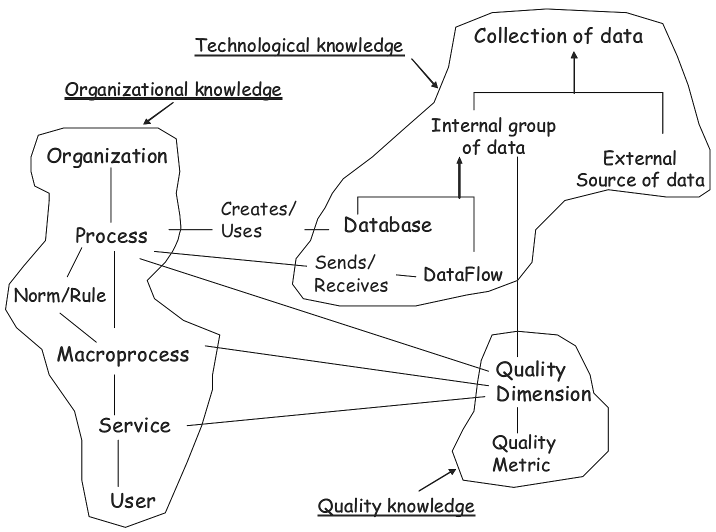


Fig. 7.1. Knowledge involved in the DQ measurement and improvement process

The main types of knowledge are

1. The *organization* or the set of organizations involved in the processes, with related organizational structures, functions, norms, and rules.
2. The business *processes* performed in the organization, and the *macroprocesses*, i.e. the groups of processes that executed together produce services or goods for users, customers and businesses.
3. The *services* delivered by processes, and the *users* requesting services.
4. The *norms/rules* that discipline the execution of processes and macroprocesses.
5. The *quality of processes, macroprocesses and services*, e.g. the execution time of a process, the usability of a service, and the accuracy of information provided by a data service.
6. The *collections of data*, corresponding to all databases and data flows which are of some interest to the organization. We distinguish among groups of data internal to the organization and external sources of data. To execute processes, organizations have to permanently store data in databases, and in order to cooperate, they have to exchange data through data flows. Both types of data, "motionless" data and "moving" data, have to be considered, since
   - errors can affect and be propagated by both; and

  - depending on their quality, they can positively or negatively influence the quality of processes.

7. The *external sources of data*, often more critical than internal data for their data quality, since there is little or no control over their production process and previous origin.
8. The *data quality dimensions* and corresponding *metrics* defined in Chapter 2, a large set is concerned with the improvement process.

Besides the types of knowledge described, other relevant elements involved in a DQ methodology are

- The *data quality activities*, which is the whole set of activities introduced in Chapter 4 that can be performed to improve the quality of data.
- *Costs and benefits* of data and of processes, regarding three different cost categories: (i) costs associated with processes due to poor data quality, (ii) costs of the improvement process, (iii) and benefits (savings and/or increased revenues) resulting from the use of better quality data. Costs and benefits have been classified in Chapter 4.

Based on the knowledge involved in the DQ measurement and improvement process, the input/output structure of a general-purpose methodology for DQ is shown in Figure 7.2.

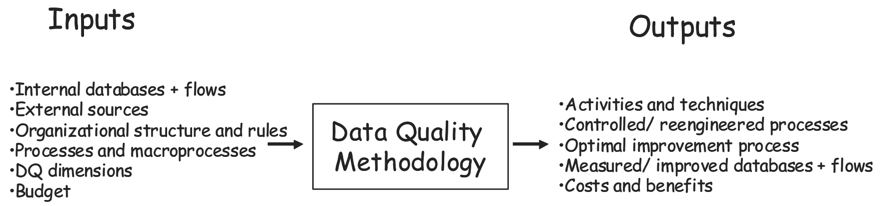

Inputs
Outputs
•Internal databases + flows
•External sources
•Organizational structure and rules
•Processes and macroprocesses
•DQ dimensions
•Budget
Data Quality Methodology
•Activities and techniques
•Controlled/ reengineered processes
•Optimal improvement process
•Measured/ improved databases + flows
•Costs and benefits

Fig. 7.2. Inputs and outputs of a DQ measurement and improvement methodology

Inputs refer to all types of knowledge described in Figure 7.1, plus the available budget, if known. The outputs concern (i) the data activities to be performed and the techniques to be applied; (ii) the business processes that have to be controlled and/or reengineered; (iii) the optimal improvement process, i.e., the sequence of activities that achieves the target quality dimensions with the minimum cost; (iv) the databases and data flows respecting new target quality dimensions; and (v) costs and benefits.

7.1.2 Classification of Methodologies

Data quality methodologies may be classified according to several criteria:

1. *Data-driven vs. process-driven.* This classification is related to the general strategy chosen for the improvement process. *Data-driven* strategies are based on using data sources exclusively to improve the quality of data; they make use of the data quality activities introduced in Chapter 4. In *process-driven* strategies, the data production process is analyzed and possibly modified to identify and remove the root causes of quality problems. We analyze this classification in more detail in Section 7.1.3. As we will see in Section 7.3, general purpose methodologies may adopt both data-driven and process-driven strategies, with different depth according to the specific methodology.
2. *Measurement* vs. *improvement.* Methodologies are needed for measuring/assessing the quality of data, or to improve their quality. Measurement and improvement activities are closely interrelated, since only when DQ measurements are available, is it possible to conceive techniques to be applied and priorities to be established. As a consequence, the boundary between the methodologies for measurement and improvement is sometimes vague. In the following, we will use the term *measurement* when we address the issue of measuring the values of a set of data quality dimensions in a database (or a set of databases). We use the term *assessment* or *benchmarking* when such measurements are compared to reference values, to enable a diagnosis of the quality of the database. Assessment methodologies will be discussed in Section 7.2.
3. *General-purpose vs. special-purpose.* A *general-purpose* methodology covers a wide spectrum of phases, dimensions, and activities, while a *special purpose* methodology is focused on a specific activity (e.g., measurement, object identification), on a specific data domain (e.g., a census, a registry of addresses of persons), or specific application domains (e.g., biology). Three of the most relevant general-purpose methodologies will be discussed in Section 7.4.
4. *Intraorganizational vs. interorganizational.* The measurement and improvement activity concerns a specific organization, or a specific sector of the organization, or even a specific process or database. Otherwise, it concerns a group of organizations (e.g., a group of public agencies) cooperating for a common goal (e.g., in the case of public agencies, providing better services to citizens and businesses).

7.1.3 Comparison among Data-driven and Process-driven Strategies

In this section we compare data-driven and process-driven strategies. We distinguish for simplicity three major strategies, using three distinct data quality activities:

1. New data acquisition from the real world. When data representing a certain reality of interest are inaccurate, incomplete or out-of-date, a possible way for improving their quality may be to again observing the reality of interest, and performing the activity called in Chapter 4 *new data acquisition.* E.g., if in a registry of employees the `DateOfBirth` is known only in 30% of the cases, we may request employees missing data. Intuitively, if the data acquisition campaign is performed effectively, this strategy immediately improves certain quality dimensions such as completeness, accuracy, and currency, since the data exactly represent the most recent reality of interest; we however note, that errors can be introduced by the measurement activity.
2. Record matching or more generally, the comparison of data whose quality dimensions have to be improved with other data in which the quality is known to be good. As an example, let us consider a database of addresses of clients that have been collected for a long period of time in a supermarket through forms, in order to provide clients with a fidelity card. After a while, certain quality dimensions, such as accuracy of residence addresses, tend to worsen. We may decide to perform a record matching activity to compare client records with an administrative database, known to be updated with the most recent data.
3. Use of data edits/integrity constraints, in which: (i) we define a set of integrity constraints against which data have to be checked, (ii) we discover inconsistencies among data, and (iii) we correct the inconsistent data by means of error localization and correction activities.

Process-driven strategies focus on processes. Consequently, they need to acquire knowledge from databases and data flows in inputs only to a limited extent. Conversely, they focus mainly on measuring the quality of processes and formulating proposals for process improvement. Two main phases characterize process-driven strategies:

- *Process control*, which inserts checks and control procedures into the data production process when (i) new data is inserted from internal or external sources, (ii) data sources accessed by the process are updated, or (iii) new data sources are involved in the process. In this way, a reactive strategy is applied to data modification events, to avoid data degradation and error propagation.
- *Process re-design*, where we avoid improving the actual process. We redesign the production processes in order to remove the causes of bad quality and introduce new activities that produce data of better quality. In the case in which the change in the process is radical, this strategy corresponds to the activity called *business process reengineering* (see [91] and [181] for a comprehensive introduction to this issue).

We compare now data- and process-driven strategies according to two coordinates of analysis: (i) the improvement the strategy is potentially able to produce on quality dimensions and (ii) the cost of its implementation. This comparison can be performed both in the short term and in the long term. In the following (see Figure 7.3), we compare improvement and costs in the long term; optimal target objectives are high improvement and low cost.

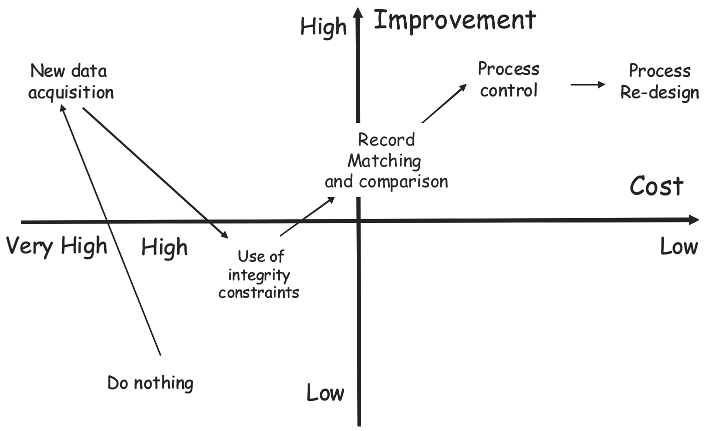


Fig. 7.3. Improvement and cost of data/process-driven strategies: comparison in the long term

The simplest and most trivial strategy is to *do nothing*. In this case, data are neglected and abandoned; certain quality dimensions, such as completeness and currency, tend to worsen in the long term. The consequence is that data progressively deteriorate the quality of business processes and the cost of lost quality increases over time.

A better strategy is *new data acquisition*; in the short term, the improvement is relevant, since data is current, complete, and accurate. However, as time goes by we are obliged to periodically repeat the process, and the cost becomes intolerable.

The strategy that uses integrity constraints leads to much lower costs, but at the same time it is less effective, since only the errors related to constraints can be checked. The errors can be corrected only to a certain extent, as we have seen in Chapter 4.

The strategy performing *record matching* has even lower costs and even more improvements, since many techniques have been developed and implemented, as we saw in Chapter 5. A relevant part of the work can be done automatically. Furthermore, once the records corresponding to the same object have been identified, high quality values can be chosen for the different attributes from the higher quality source.

In order to be effective, previous strategies that belong to the class of data driven strategies have to be repeated, leading to costs that increase in the long term. Only when we move to process-driven methods, can we optimize at the same time effectiveness and costs: *process control* activities and, above all, *process re-design* activities can get to the root of the problem and solve the problem once for all. Their costs are mainly the fixed costs related to the one shot control or re-design activity, plus variable process maintenance costs distributed over a time period.

The above considerations are valid for the long term. For the short term it is well known that process re-design can be very costly. As a consequence, data-driven strategies become more competitive. We refer the reader to [167] for a complete discussion on these issues.

7.2 Assessment Methodologies

The goal of assessment methodologies is to provide a precise evaluation and diagnosis of the state of the information system with regard to DQ issues. Therefore, the principal outputs of assessment methodologies are (i) measurements of the quality of data bases and data flows, (ii) costs to the organization due to the present low quality, and (iii) a comparison with DQ levels considered acceptable from experience, or else a benchmarking with best practices, together with suggestions for improvements. The usual process followed in assessment methodologies has three main activities:

1. relevant dimensions and metrics are initially chosen, classified, and measured;
2. subjective judgments of experts are performed; and
3. objective measurements and subjective judgements are compared.

Some examples of methodologies for the choice of dimensions and measures and for the objective vs subjective evaluation are given by Lee et al. [114], Kahn et al. in [107] Pipino et al. [161], Su et al. [185], and De Amicis et al. [56]. With regard to dimension classification, dimensions are classified in [114] (see Figure 7.4) into *sound*, *useful*, *dependable*, and *usable*, according to their positioning in quadrants related to "product quality/service quality" and "conforms to specifications/meets or exceeds consumer expectations" coordinates. The goal of the classification is to provide a context for each individual DQ dimension and metric, and for consequent evaluation. In the following we describe the methodology proposed in [56] in detail, which was tailored for the financial domain (see the main phases in Figure 7.5). For an example of benchmarking in the financial domain, see [127]. Here, we adopt the statistical term *variable* for attributes whose quality is to be measured.

Phase 1, *variables selection*, concerns the identification, description and classification of primary variables of financial registries, which correspond to the main data attributes to be assessed. The most relevant variables in financial databases are identified. Then, they are characterized, according to their meaning and role. The possible characterizations are *qualitative/categorical*, *quantitative/numerical*, and *date/time*.

	Conforms to specifications	Meets or exceeds consumer expectations
Product quality	Sound Dimensions: Free of error Coincise representation Completeness Consistent representation	Useful Dimensions: Appropriate amount Relevancy Understandability Interpretability Objectivity
Service quality	Dependable Dimensions: Timeliness Security	Usable Dimensions: Believability Accessibility Ease of operation Reputation

Fig. 7.4. Classification of dimensions in [114] for assessment purposes

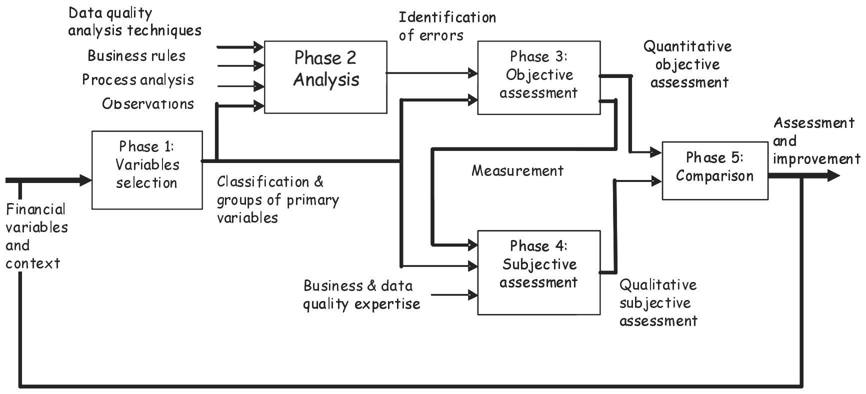

Data quality analysis techniques
Business rules
Process analysis
Observations
Phase 2 Analysis
Identification of errors
Phase 3: Objective assessment
Quantitative objective assessment
Assessment and improvement
Phase 1: Variables selection
Classification & groups of primary variables
Measurement
Phase 5: Comparison
Financial variables and context
Phase 4: Subjective assessment
Business & data quality expertise
Qualitative subjective assessment

Fig. 7.5. The main phases of the assessment methodology described in [56]

In phase 2, *analysis*, data dimensions and integrity constraints to be measured are identified. Simple statistical techniques are used for the inspection of financial data. Selection and inspection of dimensions is related to process analysis. It has the final goal of discovering the main causes of erroneous data, such as unstructured and uncontrolled data loading and data updating processes. The result of the analysis on selected dimensions leads to a report with the identification of the errors.

In phase 3, *objective/quantitative assessment*, appropriate indices are defined for the evaluation and quantification of the global data quality level. The number of erroneous observations for the different dimensions and the different data attributes is first evaluated with statistical and/or empirical methods, and, subsequently, normalized and summarized. An example of quantitative assessment is shown in Figure 7.6, where the three variables considered, typical of the financial domain are

1. Moody's rating. Moody's Investors Service is a leading provider of risk analysis, offering a system of ratings of the relative creditworthiness of securities.
2. Standard and Poor's rating, from another leading provider.
3. Market currency code, e.g. EUR.

The values associated with quality dimensions represent the percentages of erroneous data by data quality dimension. Internal consistency refers to the consistency of a data value item within the same set of financial data; external consistency refers to the consistency of a data value item in different data sets.

	Variables		
Quality dimensions	Moody's Rating	Standard's & Poor Rating	Market Currency Code
Syntactic Accuracy	1.7	1.5	2.1
Semantic Accuracy	0	0.1	1.4
Internal Consistency	2.7	3.2	1.3
External Consistency	1.6	1.1	0.1
Incompleteness	3.5	5.5	8.1
Currency	0	0	0
Timeliness	8.6	9.2	2
Uniqueness	4.9	4.9	9.3
Total (average)	3.6	3.2	3.0

Fig. 7.6. Example of objective quantitative assessment

Phase 4 deals with *subjective/qualitative assessment*. The qualitative assessment is obtained by merging three independent evaluations from (i) a business expert, who analyzes data from a business process point of view; (ii) a financial operator (e.g., a trader), who uses daily financial data; and (iii) a data quality expert, who has the role of analyzing data and examining its quality. See Figure 7.7 for a possible result of this phase, where domain values are `High`, `Medium`, and `Low`.

	Rating Moody's	Rating S&P	Market Currency Code
Syntactic Accuracy	H	H	H
Semantic Accuracy	H	H	M
Internal Consistency	H	H	H
External Consistency	H	H	M
Incompleteness	L	L	L
Currency	H	H	H
Timeliness	M	M	H
Uniqueness	H	H	H
Total	H	H	H

Fig. 7.7. Example of subjective quantitative assessment

Finally, a comparison between objective and subjective assessment is performed. For each variable and quality dimension, we calculate the distance between the percentages of erroneous observations obtained from quantitative analysis, mapped in the discrete domain `[High, Medium, Low]`, and the quality level defined by the judgment of the three experts. Discrepancies are analyzed by the data quality expert, to detect causes of errors and to find alternative solutions to correct them.

7.3 Comparative Analysis of General-purpose Methodologies

In this section we illustrate three of the most important general purpose methodologies for DQ measurement and improvement proposed in the literature and used in practice. The three methodologies are described in the literature with very heterogeneous styles and detail levels. We first describe in Section 7.3.1 using a common reference terminology the whole set of measurement and improvement steps of methodologies. Then we discuss the methodologies using such common reference terminology. The methodologies are

1. The Total Data Quality Methodology (TDQM) (see [177]), initially conceived as a research activity and subsequently widely used in several application domains.
2. The Total Quality data Methodology (TQdM), described in [68], was devised for consultancy purposes and is particularly suited for managers. The TQdM methodology has subsequently been renamed Total Information Quality Methodology (TIQM).
3. A methodology developed in the context of an Italian project, conceived by the Italian National Bureau of Census (Istituto Nazionale di Statistica, whose acronym is Istat) and the Authority for Information Technologies in Public Administration. The methodology, called in the following Istat methodology, concerns inter-organizational information systems; it was conceived for the public administration domain, and was first specialized for address data (see [74]).

Other methodologies have been proposed and are currently used. [167] describes a significant number of guidelines and experiences to be applied in DQ projects; they will not be discussed as a distinct methodology. [122] presents a methodology implemented at the Canadian Institute for Health Information (CIHI) to evaluate and improve the data quality of its administrative databases. Each database is evaluated annually against more than 80 measurable metrics within a hierarchical framework. For instance, accuracy is evaluated against 11 characteristics and 41 corresponding metrics. [103] is worth to be mentioned as a methodology for building data warehouses considering data quality aspects; the methodology adapts the Goal-Question-Metric approach from software quality management to a data management environment.

7.3.1 Basic Common Phases Among Methodologies

Basic common *phases* can be obtained by abstracting from specific notations adopted in the approaches. We distinguish between assessment and improvement processes. Common phases for the assessment process are

- *Data analysis*, which collects knowledge on data, their architecture, data flows, and data management rules. This may be achieved examining available documentation on data and logical schemas or through interviews.
- *DQ requirements analysis*, which collects general suggestions on possible causes of errors from users and data managers, and determines future targets to be achieved for data quality.
- *Find critical areas*, which chooses the most relevant databases and data flows, or their parts, to be analyzed in detail.
- *Model the process*, describes the process (or processes) according to a formal or semiformal model.
- *Perform measurement*, which establishes quality dimensions and perform measurements on the whole database or, if unfeasible or too expensive, on a sample.
- *Non-quality cost evaluation*, which estimates costs of processes due to poor data quality.
- *Benefits evaluation*, which estimates savings, increased revenues, and/or new intangible benefits deriving from the possible increase of data quality.
- *Assign responsibilities on processes*, which finds process owners for each process, and assigns them responsibilities for data production activities.
- *Assign responsibilities on data*, which finds data owners for each type of data, and assigns them responsibilities on data control.
- *Choose tools and techniques*, which chooses the most suitable tools and techniques from among available ones for the given organizational context, the domain knowledge available, and the budget.

Each of the above activities can be performed as a global step on the whole set of organizational units of an interorganizational information system, and as a specific step performed autonomously by a local organizational unit in an intraorganizational information system.

Common phases for the improvement process are:

- *Find causes of errors*, which analyzes possible causes of the deviation for each relevant deviation of quality dimensions from target values.
- *Design improvement solutions on data*, which chooses, among the DQ activities and techniques the most effective ones to be performed to achieve the targets.

- *Establish process control*, which defines checkpoints in the data production process, that allow for the monitoring and restoring of the desired quality dimensions during process execution.
- *Design improvement solutions on processes*, which besides controlling activities devised in the previous phase, finds further improvements for the steps of the actual process that produce corresponding DQ improvements.
- *Re-design processes*, finds radical changes to processes that correspondingly lead to significant data quality improvement.
- *Manage improvement solutions*. Under an organizational perspective, managers have to find new organizational rules for data quality. Such rules extend well-known quality principles for manufacturing products to data quality.
- *Check effectiveness of improvements*, which establishes periodical measurement and monitoring activities that provide feedback on the effectiveness of the process and enable its dynamic tuning.

Also in the case of improvement activities, methodological phases can involve a whole organization, or a group of organizations, or a specific organizational unit.

We detail here the three selected methodologies, highlighting specific peculiarities and proposals for organizing previous phases in a coherent process. To do so, we summarize the detailed process for each of the methodologies in a table, where a two/three level itemization is used. Names adopted in the first levels are in general coherent with the terminology introduced so far, while for subtasks we adopt the specific terminologies provided in the references cited.

7.3.2 The TDQM Methodology

The TDQM methodology proposed in [177] can be seen as an extension of total quality management to data, which was originally proposed for manufacturing products. Several enrichments of TDQM have been proposed, including the languages IP-MAP and IP-UML described in Chapter 3, leading, in this second case, to a new methodology. We describe the organization in phases of the original TDQM methodology and the IP-UML methodological extension in Figure 7.8, within the common definition framework proposed in the previous section. Terminological differences for the IP-UML extension are highlighted.

The process underlying TDQM considers four phases as necessary for managing the information products: definition, measurement, analysis, and improvement. These phases are iteratively executed, thus constituting a cycle. The *definition* phase includes the identification of data quality dimensions and related requirements. The *measurement* phase produces quality metrics that provide feedback to data quality management and allow for the comparison of the effective quality with predefined quality requirements. The *analysis* phase

1. Definition
    Data quality requirements analysis (named Quality Analysis in the IP-UML extension)
2. Measurement
    Perform measurement (part of Quality Analysis in IP-UML)
3. Analysis
    Data Analysis (the same name in IP-UML)
    Model the processes (less relevant in IP-UML)
4. Improvement (Quality improvement in IP-UML)
    Design improvement solutions on data and processes (Quality verification in IP-UML)
    Re-design processes (only in IP-UML, named Quality improvement)

Fig. 7.8. TDQM description

identifies the roots of quality problems and studies their relationships. The *improvement* phase devises quality improvement activities.

Phases defined in IP-UML are data analysis, quality analysis, and quality improvement design. Quality improvement design is composed of quality verification and quality improvement. In the *data analysis* phase, information products are identified and modeled. As a second step, in the *quality analysis* phase, the quality dimensions are defined, along with the requirements on the information product and on its constituents. It distinguishes between the requirements for raw data and component data. In Figure 7.9, an example of a quality analysis model is shown, referring to quality requirements of location data of citizens. A timeliness constraint is expressed on the information product `PureLocationData`, and completeness constraints are expressed on attributes `Municipality`, `Region`, and `Area`.

The *quality verification* phase focuses on the identification of areas that are critical, and on the quality checks to be introduced in the data flows of the information production process. Finally, the *quality improvement* phase investigates a reengineering of processes aimed at improving the quality of data. An example of a quality improvement model is shown in Figure 7.10, where the process of transfer of a citizen from one to another municipality is considered. Municipality A, where the citizen transfers from, notifies the transfer event to Municipality B, where the citizen transfers to, and to all other organizations involved in such an event. In this way, location data are kept current and accurate in all databases.

The quality requirements specified in the quality analysis model are the drivers of the re-design performed in this phase. The concept of *data steward*, i.e. person, role, or organization that is responsible for data involved in the process, is introduced. In our example of Figure 7.9, the data steward of the raw data `PureLocationData` is assumed to be the Municipality A the citizen has transferred from, and therefore Municipality A is in charge of starting the event notification.

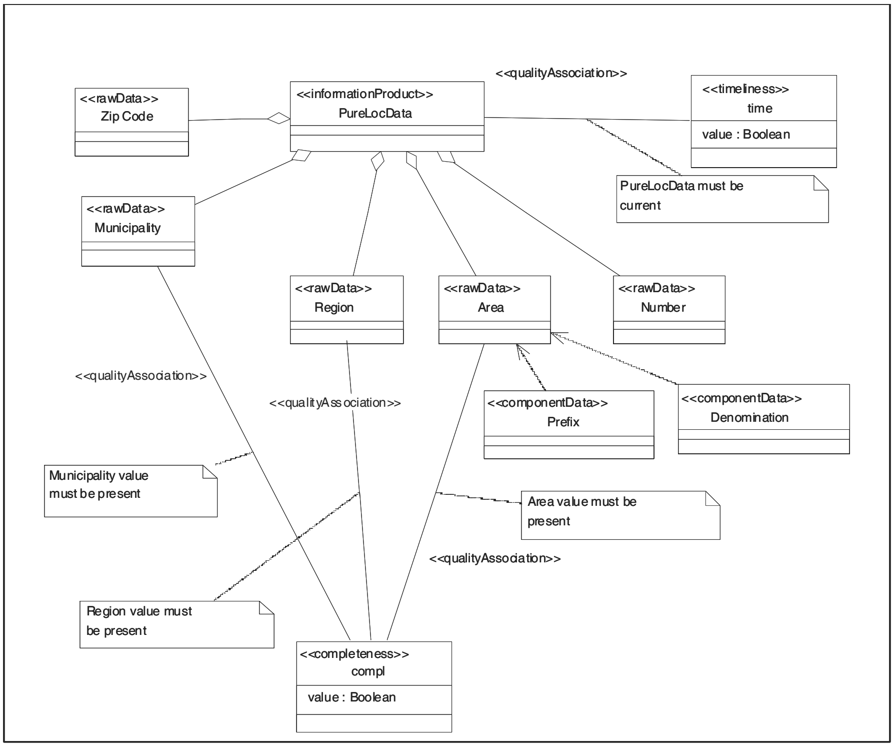


Fig. 7.9. An example of quality analysis model in IP-UML

7.3.3 The TQdM Methodology

The TQdM methodology (see [68]) was initially designed for data warehouse projects, but its broad scope and its level of detail characterize it as a general-purpose DQ methodology. In a data warehouse project, one of the most critical phases concerns the activity of off-line consolidation of operational data sources into a unique, integrated database, used in all types of aggregations to be performed. In the consolidation phase, errors and heterogeneities present in sources have to be discovered and solved, or suffer of data warehouse corruption and failure.

The orientation of TQdM toward data warehouses results in a prevalent data-driven character of the methodology. The general strategy of TQdM is synthesized in Figure 7.11. The areas in which TQdM is original and more comprehensive when compared to other methodologies are cost-benefit analysis and managerial perspective. We have discussed the cost-benefit analysis classification model of TQdM in Chapter 4. TQdM provides extensive guidelines for evaluating costs of loss of quality, costs of the process of data improvement, and benefits and savings resulting from data quality improvement. We notice here that another methodology, specifically focused on costs and

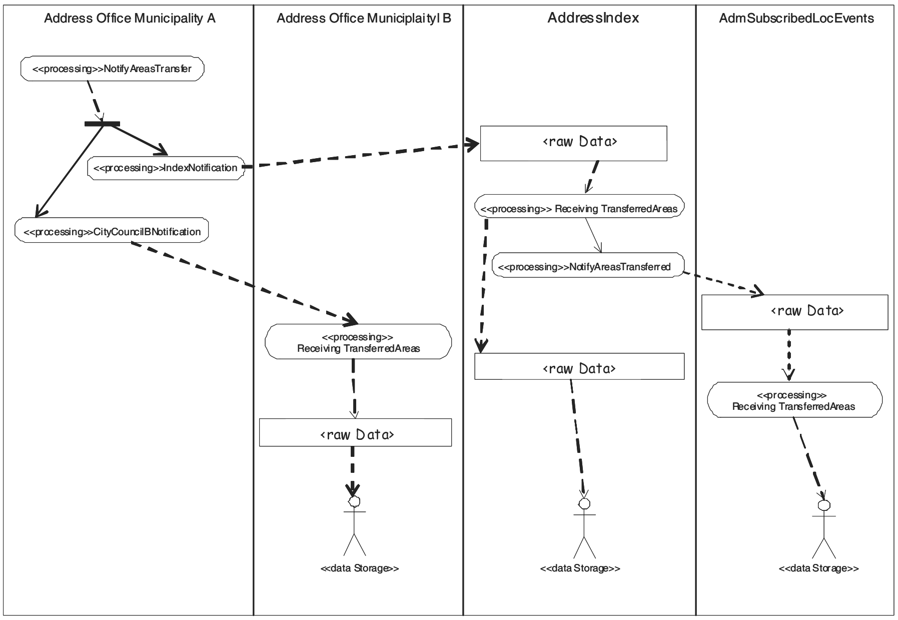

Address Office Municipality A
Address Office Municiplaityl B
AddressIndex
AdmSubscribedLocEvents
<<processing>>NotifyAreasTransfer
<<processing>>IndexNotification
<<processing>>CityCouncilBNotification
<raw Data>
<<processing>> Receiving TransferredAreas
<<processing>>NotifyAreasTransferred
<<processing>> Receiving TransferredAreas
<raw Data>
<<data Storage>>

Fig. 7.10. An example of a quality improvement model in IP-UML

savings, is described in [123], while [16] describes an integer linear programming formulation of a quality improvement process that optimizes costs. We focus now on the managerial issues of TQdM.

Management of Improvement Solutions

The main aspect discussed in TQdM concerns the managerial perspective, i.e., the strategy that has to be followed in an organization in order to make effective technical choices. The choices are in terms of DQ activities to be performed, databases and flows to be considered, and techniques adopted. So, in the final stage of TQdM, the focus is moved from technical to managerial aspects. The extent of the steps, shown in Figure 7.11, provides evidence of the attention devoted to this issue. Some steps are also present in preceding phases, which we do not comment on. Specific tasks of the managerial perspective concern:

1. Assessment of organization readiness in pursuing DQ processes.
2. Survey of customer satisfaction, in order to discover problems at the source, i.e., directly from service users.
3. Initial focus on a pilot project, in order to experiment with and tune the approach and avoid the risk of failure in the initial phase, which is typical of large-scale projects performed in one single phase. This principle is inspired by the well-known motto “think big, start small, scale fast.”

1. Assessment
   - Data analysis
     - Identify information groups and stakeholders
     - Assess consumer satisfaction
   - DQ requirements analysis
   - Measurement
     - Identify data validation sources
     - Extract random samples of data
     - Measure and intepret data quality
   - Non quality evaluation
     - Identify business performance measures
     - Calculate non quality costs
   - Benefit evaluation
     - Calculate information value
2. Improvement
   - Design solution improvement
     - On data
       - Analyse data defect types
       - Standardize data
       - Correct and complete data
       - Match, transform and consolidate data
     - On processes
       - Check effectiveness of improvement
3. Management of improvement solutions - organizational perspective
   - Assess the organization's readiness
   - Create a vision for information quality improvement
   - Conduct a customer satisfaction survey of the information stakeholders
   - Select a small and payoff area to conduct a pilot project
   - Define the business problem to be solved
   - Define the information value chain
   - Perform a baseline assessment
   - Analyze customer complaints
   - Quantify costs due to quality problems
   - Define information stewardship
   - Analyze the systematic barriers to DQ and recommend changes
   - Establish a regular mechanism of communication and education with senior managers

Fig. 7.11. TQdM description

4. Definition of information stewardship, i.e., the organizational units and their managers who, with respect to the laws (in public administrations) and rules (in private organizations) that govern business processes, have specific authority on data production and exchange.
5. Following the results of the readiness assessment, analysis of the main barriers in the organization to the DQ management perspective in terms of resistance to change processes, control establishment, information sharing, and quality certification. In principle, every manager thinks that her or his data is of very high quality, and he or she is reluctant to accept controls, respect standards and methods, and share information with other managers. This step concerns a well-known habit of managers to consider data as a type of power.
6. Establishmnet of a specific relationship with senior managers, in order to get their consensus and active participation in the process.

Before concluding this section on TQdM, we mention a second set of major managerial principles inspired by [50].

- Principle 1. Since data are never what they are supposed to be, check and recheck schema constraints and business rules every time fresh data arrive. Immediately identify and send discrepancies to responsible parties.
- Principle 2. Maintain a good and strict relationship with the data owners and data creators, to keep up with changes and to ensure a quick response to problems.
- Principle 3. Involve senior management willing to intervene in the case of uncooperative partners.
- Principle 4. Data entry, as well as other data processes, should be fully automated in such a way that data be entered only once. Furthermore, data should only be entered and processed as per schema and business specifications.
- Principle 5. Perform continuous and end-to-end audits to immediately identify discrepancies; the audits should be a routine part of data processing.
- Principle 6. Maintain an updated and accurate view of the schema and business rules; use proper software and tools to enable this.
- Principle 7. Appoint a data steward who owns the entire process and is accountable for the quality of data.
- Principle 8. Publish the data where it can be seen and used by as many users as possible, so that discrepancies are more likely to be reported.

7.3.4 The Istat Methodology

The Istat methodology (see [74] and [73]) has been designed for Italian public administration. Specifically, it concerns address data of citizens and businesses. Notwithstanding these limitations, it is characterized by a rich spectrum of strategies and techniques that allow for its adaptation to many other domains. The principal reason for this is the complexity of the structure of the Italian public administration, as of many others, characterized typically by at least three tiers of agencies:

1. *central agencies*, located close to each other, usually in the capital city of a country;
2. *peripheral agencies*, corresponding to organizational structures distributed thorough the territory, hierarchically dependant on central agencies;
3. *local agencies*, that are usually autonomous from central agencies, and correspond to districts, regions, provinces, municipalities, and other smaller administrative units. Sometimes they are functionally specialized, e.g., hospitals.

The above is an example of the organizational structure of public administration; it has many variants in different countries. The common aspects to many administrative, organizational, and technological models concern

- their complexity, in terms of interrelations, processes, and services in which they are involved, due to the fragmentation of competencies among agencies. This frequently involves information flows exchanged between several agencies at the central and local level;
- their autonomy, which makes it difficult to enforce common rules; and
- the high heterogeneity of meanings and representations that characterize databases and data flows, and the high overlapping of usually heterogeneous records and objects.

Improving DQ in such a complex structure is usually a very large and costly project, needing an activity that may last several years. In order to solve the most relevant issues related to data quality, in the Istat methodology attention is primarily focused on the most common type of data exchanged between agencies, namely, address data. When compared to previously examined methodologies, this methodology is innovative since it addresses all the coordinates introduced in Section 7.1.2, specifically, data vs process-driven, and intraorganizational vs interorganizational. A synthetic description of the Istat methodology is shown in Figure 7.12, where the three main phases are represented, together with the information flows between them.

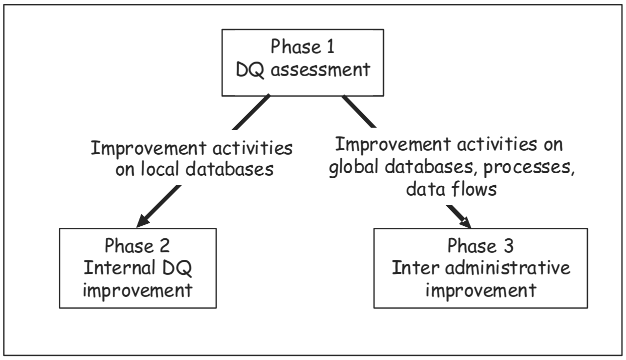
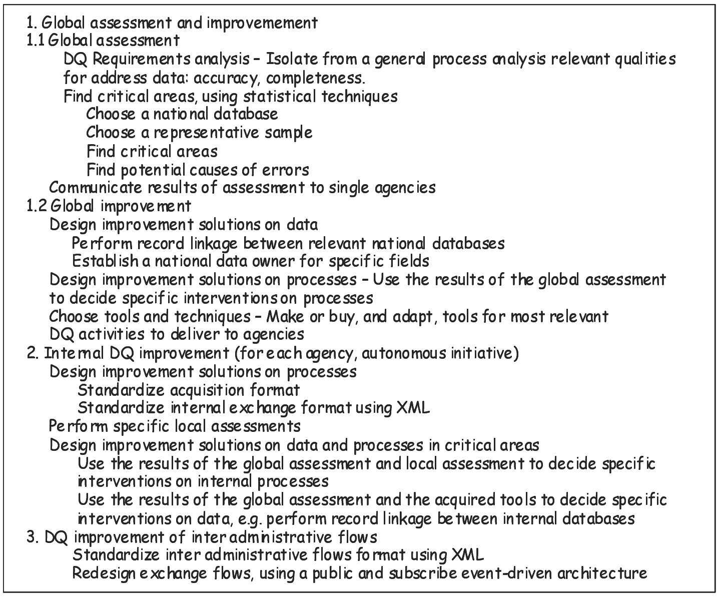

Phase 1
DQ assessment
Improvement activities on local databases
Improvement activities on global databases, processes, data flows
Phase 2
Internal DQ improvement
Phase 3
Inter administrative improvement

Fig. 7.12. General view of the Istat methodology

The assessment made in Phase 1 identifies the most relevant activities to be performed in the improvement process. These activities are:

1. Phase 2, activities on databases locally owned by agencies under their responsibility. Tools are distributed for performing these types of activities autonomously, and courses are offered for learning more on DQ issues.
2. Phase 3, activities that concern the overall cooperative information systems of administrations, in terms of exchanged data flows, and central databases set up for possible coordination purposes. These activities are centrally planned and coordinated.



1. Global assessment and improvemement
1.1 Global assessment
DQ Requirements analysis – Isolate from a general process analysis relevant qualities for address data: accuracy, completeness.
Find critical areas, using statistical techniques
Choose a national database
Choose a representative sample
Find critical areas
Find potential causes of errors
Communicate results of assessment to single agencies
1.2 Global improvement
Design improvement solutions on data
Perform record linkage between relevant national databases
Establish a national data owner for specific fields
Design improvement solutions on processes – Use the results of the global assessment to decide specific interventions on processes
Choose tools and techniques – Make or buy, and adapt, tools for most relevant DQ activities to deliver to agencies
2. Internal DQ improvement (for each agency, autonomous initiative)
Design improvement solutions on processes
Standardize acquisition format
Standardize internal exchange format using XML
Perform specific local assessments
Design improvement solutions on data and processes in critical areas
Use the results of the global assessment and local assessment to decide specific interventions on internal processes
Use the results of the global assessment and the acquired tools to decide specific interventions on data, e.g. perform record linkage between internal databases
3. DQ improvement of inter administrative flows
Standardize inter administrative flows format using XML
Redesign exchange flows, using a public and subscribe event-driven architecture

Fig. 7.13. Detailed description of the Istat methodology

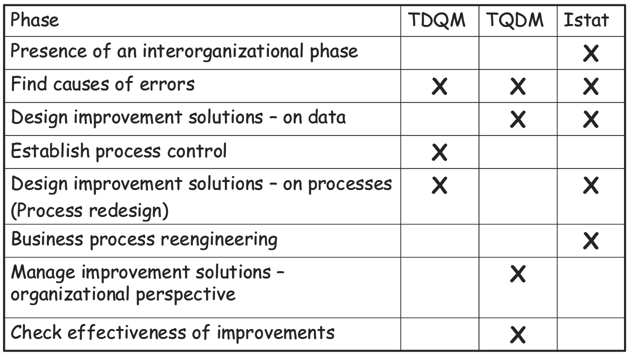
A more detailed description of the methodology is shown in Figure 7.13; the innovative aspects concern

- the assessment phase, initially performed on central databases, with the goal of detecting a priori critical areas. For example, within addresses of some regions, such as New Mexico in the US or Alto Adige in Italy, the names of streets are bilingual or they have a different spellings in their original and official languages, leading to errors. In our example, the original languages are, respectively, Spanish and German, and the official languages are English and Italian. New Mexico and Alto Adige are potentially critical areas for the assessment phase;
- the application of a variety of simple but effective statistical techniques in quality measurement steps;
- the definition of data owners at a very detailed granularity level, corresponding to single attributes, such as `MunicipalityCode` and `SocialSecurityNumber`;
- the arrangement of tools and techniques for the most relevant cleaning activities produced and distributed to single agencies, assisting them in tailoring the activities to specific territorial or functional issues;
- the standardization of address data formats and their expression in a common XML schema, implemented to minimize internal changes to agencies and to allow interoperability in flows between agencies;

- the redesign of exchanged data flows, using a publish and subscribe event-driven technological architecture, an example of which we will see in the case study at the end of the chapter.

7.3.5 Comparisons of Methodologies

In Figure 7.14, we compare the three previously described methodologies by showing the degree of coverage of improvement phases introduced in Section 7.3.1. A cross in a cell means that the phase is adequately covered in the methodology, with original strategies, techniques, and suggestions; its absence means that the phase is absent or poorly covered. A criterion based on prevalence is adopted.



Phase	TDQM	TQDM	Istat
Presence of an interorganizational phase			X
Find causes of errors	X	X	X
Design improvement solutions – on data		X	X
Establish process control	X		
Design improvement solutions – on processes (Process redesign)	X		X
Business process reengineering			X
Manage improvement solutions – organizational perspective		X	
Check effectiveness of improvements		X	

Fig. 7.14. Comparison of improvement steps mainly covered in methodologies

First of all, only the Istat methodology provides for an interorganizational approach, while TDQM and TQdM are suitable for specific organizations or information products. All three methodologies provide guidelines for finding the causes of errors. Concerning their attitude with respect to data-driven vs process-driven strategies, TDQM has a clear approach oriented to process driven guidelines, while TQdM and Istat cover both data-driven and process-driven activities, though TQdM does so to a limited extent. With regard to the type of process improvement suggested, no methodology covers the three types, namely process control, process redesign, and business process reengineering, while only Istat addresses the strategy of radically changing the processes, through process reengineering activities.

As we have already seen, TQdM is the methodology most suited to managers. It provides a large number of indications for applying and generating a consensus for the methodology in an organization. TQdM is also unique in establishing detailed guidelines for checking the effectiveness of improvements.

We compare the methodologies also with regard to the level of formalism used and the consolidation. TDQM is very rich, as we have seen in Chapter

3, in the model proposed to describe the data production process. TQdM typically uses very simple formalisms, e.g., charts. The Istat methodology provides a significant number of statistical techniques. With regard to the consolidation, TDQM and TQdM have been widely applied since the 1990s in U.S. and, to some extent, in other countries, while the Istat methodology is very recent, with a limited number case studies documented.

7.4 The CDQM methodology

Now we discuss an original methodology, characterized by a reasonable balance between completeness and the practical feasibility of the data quality improvement process. The methodology deals with all types of knowledge described in Figure 7.1; for this reason, we will call it *Complete Data Quality Methodology* (*CDQM*) methodology. The phases and steps of CDQM are shown in Figure 7.15.

Phase 1: State reconstruction

1. Reconstruct the state and meaning of most relevant databases and data flows exchanged between organizations, and build the *database + dataflow/organization matrixes.*
2. Reconstruct most relevant business processes performed by organizations, and build the *processes /organizations matrix.*
3. For each process or group of processes related in a macroprocess, reconstruct the norms and organizational rules that discipline the macroprocess and the service provided.

Phase 2: Assessment

4. Check the major problems related with the services provided with the internal and final users. Fix these drawbacks in terms of process and service qualities, and identify the causes of the drawbacks due to low data quality.
5. Identify relevant DQ dimensions and metrics, measure data quality of databases and data flows, and identify their critical areas.

Phase 3: Choice of the optimal improvement process

6. For each database and data flow, fix the new DQ levels that improve process quality and reduce costs under a required threshold.
7. Conceive process re-engineering activities and choose DQ activities, that may lead to DQ improvement targets set in step 6, relating them in the data/*activity matrix* to clusters of databases and data flows involved in DQ improvement targets.
8. Choose optimal techniques for the DQ activities.
9. Connect crossings in the *data/activity matrix* in reasonable candidate improvement processes
10. For each improvement process defined in the previous step, compute approximate costs and benefits, and choose the optimal one, checking that the overall cost-benefit balance meets the targets of step 6.

Fig. 7.15. Phases and steps of CDQM

The overall strategy of CDQM sees the measurement and improvement activities as being deeply related to the business processes and to the costs of the organization. In phase 1 all the most important relationships between organization units, processes, services, and data, if not known are reconstructed. Phase 2 sets new target quality dimensions which are needed to improve process qualities, and evaluates reduced costs and new benefits. Phase 3 finds the optimal improvement process, i.e., the sequence of activities that has the optimal cost-effectiveness. In this section we examine the specific steps. The next section will provide a detailed case study.

7.4.1 Reconstruct the State of Data

Similarly to what happens in information system planning methodologies, at the beginning of the DQ process we reconstruct a model of the most relevant relationships between organizations or organizational units and data used and exchanged. This information is important, since it provides a picture of the main uses of data, of providers, and of consumers of data flows. We can represent these relationships with two matrixes:

1. the *database/organization matrix* (see Figure 7.16), where, for the most relevant databases, we represent organizations that create data and organizations that use data. This matrix could be refined, representing single entities (or tables), but in order to make its size reasonable, we set the granularity at the database level; and

Database/ Organization	Database 1	Database 2	………	Database n
Organization 1	Creates	Uses		Uses
Organization 2		Uses		
…………				
Organziation m		Creates		Creates

Fig. 7.16. The database/organization matrix

2. the *dataflow/organization matrix* (see Figure 7.17), similar to the previous one, in which we represent the provider and consumer organizations of the most relevant data flows.

Dataflow/ Organization	Dataflow 1	Dataflow 2	………	Dataflow n
Organization 1	Provider	Consumer		Consumer
Organization 2		Consumer		Provider
…………				
Organziation m	Consumer	Provider		Consumer

Fig. 7.17. The dataflow/organization matrix

7.4.2 Reconstruct Business Processes

In this step we focus on processes and their relationships with organizational units. *Processes* are units of work performed in the organization and related to the production of goods or services. For every process we have to find the organizational unit that is its owner, and the units that participate in the execution of the process: the whole set of cross-relationships is represented in the *process/organization matrix*, an example of which is given in Figure 7.18. Distinguishing the owner of the process is important in DQ issues, since we can assign precise responsibilities in data-driven and process-driven improvement activities.

Process/ Organization	Process 1	Process 2	....	Process n
Organization 1	Owner	Participates		
Organization 2		Participates		Owner
....				
Organziation m	Participates	Owner		Participates

Fig. 7.18. The process/organization matrix

7.4.3 Reconstruct Macroprocesses and Rules

In this step we analyze two aspects in depth: the structure and the final objectives of the processes in the organization, i.e., how they are related and linked in the production of goods/services (denoted in the following for simplicity as services), and the legal and organizational rules that discipline and specify this structure. The relevant characteristics of processes are described in the *macroprocess/norm-service-process matrix* (see Figure 7.19), where the following aspects are represented:

- the *macroprocess*, i.e., the set of processes that are all together involved in service provision;
- *services* provided, identified by a name and, possibly, by the class of users of the service, their characteristics, and the organization responsible for service provision;
- *norms* that discipline the high-level specification of the process.

Reconstructing the macroprocesses is an important activity, since modeling processes independently provides only a fragmented view of the activities of the organization. On the contrary, we need an integrated view to make decisions related to the possible restructuring of processes and information flows. At the same time, especially in public organizations, the knowledge of norms related to the macroprocesses is relevant to precisely understand (i) the area at our disposal for "maneuvers" in process-driven activities, (ii) the extent to which we are free to restructure processes, and (iii) the norms or organizational rules to be repealed, changed, or modified.

Notice in Figure 7.19 that macroprocesses are represented as a set of processes. This model is very simple, and could be enriched using a process specification language (see example in [193]).

Macroprocess	Macroprocess1	Macroprocess2	....	Macroprocess m
Norm/organiza-tional rule	Norm 1	Norm 2		Norm3 and Norm4
Service(s)	S1 and S5	S2 and S5		S3 and S4
Process 1	X			
Process 2		X		
Process 3	X			
Process 4	X			
...				
Process n				X

Fig. 7.19. The macroprocess/norm-service-process matrix

7.4.4 Check Problems with Users

The goal of this step is to identify the most relevant problems, in terms of causes of poor data quality. Focusing initially on services, they can be identified by interviewing internal and final users, and by understanding the major burdens and negative effects of poor data quality on the activities of internal users and on the satisfaction of final users. Then, the analysis goes back to processes to find the causes, in terms of quality and the nature of processes, that produce such burdens and negative effects. As an example, taxpayers of a district are bothered, if they receive erroneous notices of assessment from the revenue agency. It may be discovered that tax files for that district are not accurate, due to delayed or incorrect updates.

7.4.5 Measure Data Quality

In previous steps we have identified main problems that lead to poor data quality; here, we have to select, among the set of dimensions and metrics discussed in Chapter 2, the most relevant ones for our problem; for such dimensions, we have to choose metrics to provide a quantitative evaluation of the state of the system. For example, if the major burden perceived by final users is the time delay between an information service request and service provision, we have to focus on the currency dimension, and organize a process to measure it.

Another relevant aspect of this step is locating critical areas, mentioned in the discussion on the Istat methodology. Since the improvement activities are complex and costly, it is advisable to focus on the parts of databases and data flows that reveal major problems. This activity can be performed in two ways:

- Analyzing problems and causes, and trying to identify the data whose poor quality is more negatively influenced by them. In the taxpayer example, we focus on one specific district, since complaints come prevalently from that area.
- Analyzing statistics on data quality metrics selected according to different properties of data, and determining where poor quality is located. We have seen this case in the example on names of streets discussed in Section 7.3.4.

7.4.6 Set New Target DQ Levels

In this step we set new target DQ levels, evaluating the economic impact of the improvement in terms of (hopefully) reduced costs and improved benefits. We have discussed in Chapter 4 some classifications of costs and benefits, and proposed a new one. The idea in this step is to use such classifications as a checklist; for each item in the classification, or in a subset of it, we collect data that allow some approximate estimate of the costs, savings, and other benefits associated with the item. Some items are easily calculated, such as, e.g., the cost of equipment involved in data cleaning activities. Other items need an estimate. For example, we may have perceived that a significant cost item is related to the time spent by clerks in looking for unmatched citizens, or for missing businesses in a registry. In the former case, we (i) estimate the number of clerks involved in the activity in terms of person-months a year; (ii) multiply this number by the average of the gross salary. Some cost items are difficult or even impossible to estimate. In this case, we identify a proxy cost item that provides an indirect valuation of the item that cannot be estimated.

Other aspects to be addressed concern the so called *intangible benefits*, which are difficult to express in monetary terms, and have to be eventually considered on a qualitative basis. Finally, the calculation of return on investment is useful to help senior management make a decision about the level of commitment to the data quality program.

The last issue to be dealt with in this step is the establishment of a relationship between costs, benefits, and quality levels. For instance, we assume that presently 10% of customer addresses are not correct, and such poor quality reduces potential revenues of sales campaigns by 5%. We have to identify, at least qualitatively, the functions that relate(i) costs of processes, (ii) savings, and (iii) the cost of the improvement program for accurate addresses. Then, we have to superimpose the three functions, to find the optimal balance between cost and savings and the corresponding target quality level to be achieved.

7.4.7 Choose Improvement Activities

This step is perhaps the most critical one for the success of the methodology. The goal here is to understand which process-driven activities and which data-driven activities lead to the most effective results for quality improvement of databases and data flows. In this choice we may group databases and data flows or split them, in order to examine only critical areas or specific parts that are relevant in an activity.

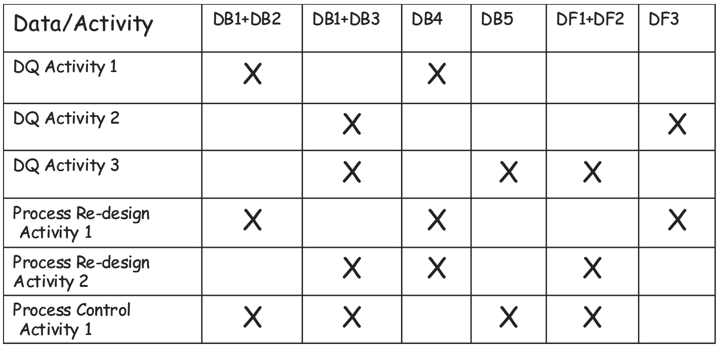
With regard to process-driven activities, business process reengineering activity (see [91], [181], and [137] for a comprehensive discussion) is composed of the following steps:

- Map and analyze the *as-is process*, in which the objective is typically to describe the actual process.
- Design the *to-be process*, producing one or more alternatives to the current process.
- Implement a reengineered process and improve continuously.

Data-driven activities have been described in great detail in previous chapters. To choose from them, we have to start the analysis from causes and problems, discovered in step 4. We discuss a few cases.

1. If a relational table has low accuracy, and another source represents the same objects and common attributes with higher accuracy, we perform an *object identification* activity on the table and the source. Then, we select the second source for values of common attributes.
2. Assume that a table exists used mainly for statistical applications, and characterized by low completeness. We perform an *error correction* activity that changes null values to valid values, keeping the statistical distribution of values unchanged.
3. Assume that a certain data flow is of very poor quality; in this case, we perform a source selection activity on data conveyed by the data flow. The goal of a *source selection* activity is to change the actual source, selecting one or more data sources that together provide the requested data with better quality. Source selection can be seen as a particular case of quality-driven query processing, discussed in Chapter 6.

At the end of the step, we should be able to produce a *data/activity matrix* like the one shown in Figure 7.20, where we put a cross for every pair of (i) activity and (ii) groups of databases or data flows to which it applies.



Data/Activity	DB1+DB2	DB1+DB3	DB4	DB5	DF1+DF2	DF3
DQ Activity 1	X		X			
DQ Activity 2		X				X
DQ Activity 3		X		X	X	
Process Re-design Activity 1	X		X			X
Process Re-design Activity 2		X	X		X	
Process Control Activity 1	X	X		X	X	

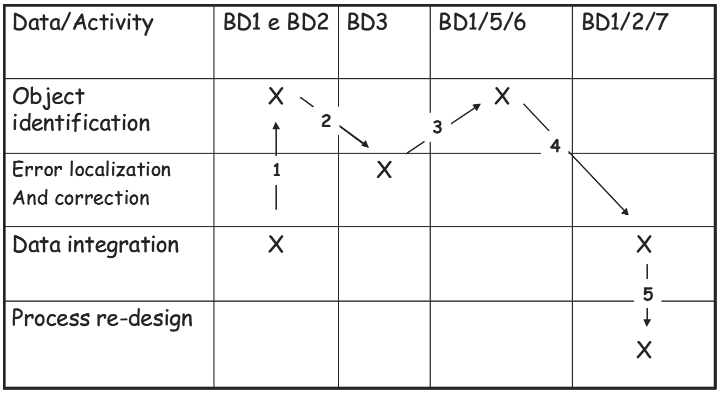
Fig. 7.20. The data/activity matrix

7.4.8 Choose Techniques for Data Activities

In this step we have to choose the best technique and tool for each data activity in the *data/activity matrix*. To choose the technique, starting from the available knowledge domain, we use all the arguments and comparative analysis dealt with in Chapters 4, 5, and 6. Here, we need to look at the market to check which techniques, among the chosen ones, are implemented in commercial DQ tools. We have to compare their costs and technical characteristics; therefore, the choice of the technique is influenced by the market availability of the tools. With reference to the object identification activity, many commercial tools or open source tools include probabilistic techniques, while tools adopting empirical and knowledge-based techniques are less widespread. If the tool is extendible, it can be chosen and then adapted to specific requirements. For instance, assume that we have performed in the past a deduplication activity on citizens of a country, in which last names are typically very long; now we have to perform the same activity on citizens of another country where last names are shorter. If in the past we have used a probabilistic technique with given distance functions for the attributes `Name`, `LastName`, and `Address`, we could modify the technique, adapting the decision procedure to the changed context, by changing, for instance, for the attribute `LastName` the distance function and weights as discussed in Chapter 5.

7.4.9 Find Improvement Processes

We now have to link crosses in the data/activity matrix in order to produce possible candidate improvement processes, with the objective of achieving completeness, i.e., all databases and data flows involved in the improvement program are covered. Linking of crossings in the data/activity matrix can be performed in several ways, and gives rise to several candidate processes, two or three of them usually sufficient to cover all possible relevant choices. In Figure 7.21 we see one of them, in a context in which we have chosen object identification, error correction, and data integration as data-driven activities, and business process reengineering as process-driven.



Data/Activity	BD1 e BD2	BD3	BD1/5/6	BD1/2/7
Object identification	X		X	
Error localization And correction		X		
Data integration	X			X
Process re-design				X

Fig. 7.21. An example of improvement process

7.4.10 Choose the Optimal Improvement Process

We are close to the solution; we now have to compare the candidate improvement processes from the point of view of the cost of the improvement program. For instance, anticipating a business process reengineering activity may lead to a more efficient object identification activity, and anticipating an object identification activity results in simpler error correction.

Items to be considered in cost evaluation include cost of equipment, cost of personnel, cost of licenses for tools and techniques, and cost of new custom software to be realized for ad hoc problems. Once the costs are evaluated and compared, we choose the most effective improvement process. At this point, it is important to compare again the costs of the selected improvement process with net savings (hopefully) resulting from the set new DQ levels step; the net final balance should be positive; otherwise, it is better to do nothing!

7.5 A Case Study in the e-Government Area

In this section we apply CDQM to a real-life case study, described in detail in [4], typical of Government-to-Business relationships in many countries. Businesses, in their life cycle have to interact with several agencies to request administrative services. The interactions are needed for several business events. Examples of such events and related services are

- starting a new business or closing down a business, which involves registering the business, e.g., with the chamber of commerce;

- evolving a business, which includes variations in legal status, board composition and senior management, number of employees, as well as the launching of a new location, and the filing for a patent;
- other services concern territorial marketing, i.e., providing thematic information on the territory in order to facilitate the creation of business networks and extend product markets; and
- security (e.g., issue of smart cards for service access, authentication, and authorization) and general enquiry services used by businesses.

In their interaction with businesses, agencies manage both agency-specific information, such as employee social insurance taxes, tax reports, balance sheets, and information common to all the businesses, typically including the following:

- attributes that characterize the business, including one or more identifiers, headquarters and branch addresses, legal structure, main economic activity, number of employees and contractors, and information about the owners or partners;
- milestone dates, including date of business start-up and date of cessation.

Each agency usually makes different use of pieces of the common information. As a consequence, each agency enforces different types of quality control, that are deemed adequate for local use of the information. Since every business reports independently to each agency, the copies have different levels of data accuracy and currency. As a consequence, similar information about one business is likely to appear in multiple databases, each autonomously managed by different agencies that historically have never been able to share their data about the businesses. The problem is aggravated by the many errors contained in databases, that cause mismatches between the different records that refer to the same business. One major consequence of having multiple disconnected views for the same information is that businesses experience severe service degradation during their interaction with the agencies.

Because of the above complications mentioned, a project is launched that follows two main strategies, aimed at improving the state of existing business data and at maintaining correct record alignment for all future data:

1. Extensive record matching and data cleaning should be performed on existing business information, resulting in the reconciliation of a large amount of business registry entries.
2. A "one-stop shop" approach is followed to simplify the life of a business and to ensure the correct propagation of its data. In this approach, a single agency is selected as a front-end for all communication with the businesses. Once the information received by a business is certified, it is made available to other interested agencies through a publish/subscribe event-driven infrastructure.

Now we apply CDQM assuming for simplicity that we deal with three agencies, namely the Social Security agency, the Accident Insurance agency, and Chambers of commerce.

Reconstruct the State of Data

In Figures 7.22 and 7.23 we report the present situation of the databases managed by the three agencies, and data flows between agencies and businesses. Each agency has its own registry of businesses; no shared database exists. Concerning flows, each agency receives information from businesses for service requests, and sends back to businesses information related to service provision.

Database/ Organization	SocialSecurity Registry of businesses	Accident Insurance Registry of businesses	Chambers of Commerce Registry of businesses
SocialSecurity	Creates/Uses		
Accident Insurance		Creates/Uses	
Chambers of Commerce			Creates/Uses

Fig. 7.22. The database/organization matrix

Dataflow/ Organization	Dataflow 1: Information for service request	Dataflow 2: Information related to service provision
SocialSecurity	Consumer	Provider
Accident Insurance	Consumer	Provider
Chambers of Commerce	Consumer	Provider
Businesses	Provider	Consumer

Fig. 7.23. The dataflow/organization matrix

Reconstruct Business Processes

We focus on interactions between businesses and agencies where businesses have to inform agencies of a large set of variations in their status according to existing administrative rules. This covers change of address of the registered office, headquarters, and branches, and updates to main economic activity. In Figure 7.24 we show three of these processes that have the common feature of involving (in distinct threads) all three agencies. As evident from the figure, coordination does not presently exists between agencies in the management of common information.

Process/ Organization	Update registered office info	Update branches info	Update main economic activity info
SocialSecurity	X	X	X
Accident Insurance	X	X	X
Chambers of Commerce	X	X	X

Fig. 7.24. The process/organization matrix

Reconstruct Macroprocesses and Rules

We assume that every interaction between a business and an agency that informs the agency of a variation of status, is ordered by a law or as more frequent by organizational rules specific to each agency. Examples of these rules are

1. the business can be represented by an agent, but in this case the agent should have been accredited in advance by the agency;
2. when the update is made a specific form has to be used;
3. the agency has to be informed of the variation within 60 days after the corresponding event.

With regard to macroprocesses, as we stated we assume a very fragmented situation of administrative activities, in which interactions with businesses are completely independent of each other. In this case, macroprocesses consist of the chain of activities related to the update, which consists of (i) entering information into the database, (ii) if necessary, providing a receipt to the business or intermediary, (iii) and sending a message to the business if inconsistencies have occurred.

Other processes concern, for example, the payment of pensions or insurance contributions. In some countries, they are deducted from wages and paid directly by businesses. For these processes the macroprocess is much more complex. It includes transactional activities such as collection and registration of payments, correctness checks, and other related processes such as discovery of and contribution evasion recovery.

Check Problems with Users

We have now to interact with the internal and final users of the data and analyze their perception of the quality of data they use (internal users) or get from the agencies. We assume that the results of interviews can be summarized as follows.

1. Internal users are frustrated by the fact that businesses contacted frequently complain about multiple letters, messages, or telephone calls. This is a sign of the presence of duplicate objects in the databases.
2. Internal users involved in tax frauds do not succeed in matching businesses when they perform cross-queries on several databases. For example, taxes paid and energy consumption are not found among the three databases of agencies in cross-queries searching for tax evaders. This is an indication of loose matching of records in databases.
3. Final users (businesses) contacted by phone interviews are burdened by the fact that for a long time after the communication of variations, e.g., of the address ("several months" is typical), they do not receive letters or messages from agencies at the new address. Conversely internal users receive a huge amount of messages back from addresses that correspond to unknown businesses. This in an indication of the lengthy period it takes to perform updates in the database.
4. Final users are very unhappy about the long lines at counters, the time lost in providing variation information, and the long delays in administrative procedures.

From the results of interviews, and a qualitative analysis of processes described in step 2, we conclude that we have to focus on the following quality dimensions and metrics:

- presence of duplicate objects in single databases, classified in Chapter 2 as inaccuracy;
- presence of non-matching objects in the three databases, again classified as inaccuracy;
- delay in the registration of updates, a case of low currency.

Apart from accuracy and currency, other quality dimensions, e.g., completeness of databases, result in similarly relevant problems. Furthermore, we could consider also the quality resulting from item 4 of the previous list, i.e. the burden for the business resulting from long lines, corresponds to time lost in interaction with the agency, and the service time spent by the agency; these are not data quality dimensions, but, in any case, they are important qualities that need to be improved on in the project. In a data quality improvement project, a larger set of problems and improvement objectives have to be addressed, in addition to those about the quality of data. These aspects are related to the quality of processes and the quality of the services.

Measure Data Quality

In the previous step we identified the quality dimensions to focus on. Now we have to choose related metrics, and organize a process to measure the actual values. With reference to previous dimensions,

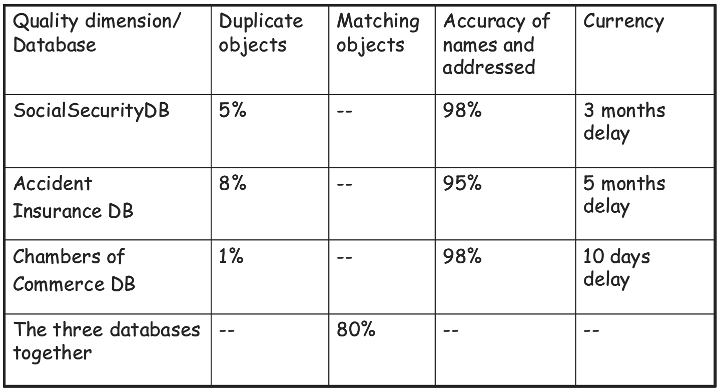
- accuracy can be measured with the percentage of duplicates and the percentage of non-matching objects;
- currency can be measured as the average delay between the time t_1, at which the information "enters" the agency, and the time t_2, at which it is registered in the system.

The measurement process for accuracy (and for completeness if considered) can be performed on a sample of the database. In the choice of samples, a set of tuples must be selected that are representative of the whole universe, and in which the overall size is manageable. Methodologies for choosing suitable samples are described in [68]. For time dimension measurements, we interview internal or final users, in order to get a better estimate of their rough perception of the delay. Otherwise, for the time spent by the agency in performing the administrative process, we make a more precise evaluation: starting from the same sample chosen for accuracy, we measure time spent as the time interval between process start and process end. This is made easy by the presence of a workflow tool that traces interaction events in input and output to and from the agency. At the end of the measurement process, we should be able to fill in the table shown in Figure 7.25.

Set New Target DQ Levels

New data quality levels have to be correlated with the desired benefits, in terms of cost savings and other measurable benefits. Cost savings estimation needs to evaluate actual costs and reduced costs due to the data quality improvement.

Two cost drivers that are a direct consequence of the misalignment can be chosen as more relevant: the heterogeneity and the poor accuracy of names and addresses at the agencies. First, we assume that agencies, conscious of the misalignment and inaccuracy of addresses, spend an estimated 10 million Euros a year to correct and reconcile records using clerical review, for example to manually trace businesses that cannot be correctly and unequivocally identified. Second, because most tax fraud prevention techniques rely on cross-referencing records over different agencies, misalignment results in



Quality dimension/ Database	Duplicate objects	Matching objects	Accuracy of names and addressed	Currency
SocialSecurityDB	5%	--	98%	3 months delay
Accident Insurance DB	8%	--	95%	5 months delay
Chambers of Commerce DB	1%	--	98%	10 days delay
The three databases together	--	80%	--	--

Fig. 7.25. Actual quality levels

undetected tax fraud; this phenomenon is made more critical by the practical impossibility of reaching businesses whose addresses are incorrect or not current. Tax fraud can be roughly estimated as a percentage, depending on the country, between 1 % and 10 % of the gross domestic product. A country with a gross domestic product equal to 200 billion Euros, assuming a (conservative) percentage of 1.5 %, has reduced revenues equal to at least 300 millions Euros.

In a broader sense, we investigate other costs involved with the low quality of processes and services. In the traditional, non-integrated setting, the burden of business transactions is shared between the businesses and the agencies. The costs to businesses, in terms of personnel involved and fees to intermediaries, can be estimated on the basis of the number of events per year. If, for example, we assume two millions events per year, and three person-hours spent for each event, we estimate a loss of 200 million Euros per year. On the agency side, the cost of handling a single transaction is about five Euros, equivalent to 20-25 person-minutes devoted to the internal bookkeeping associated with a single business event. Overall, the cost for a single agency to handle the inefficiency, considering its own events, is no less than 10 million Euros per year. Assuming that the records of each business appear in the databases of at least ten agencies, this brings the total cost per year to 100 million Euros or more.

We come to the conclusion that in order to make the use of the publish and subscribe infrastructure effective, and to reduce tax evasion with the consequence of increasing revenues, we need to set the following targets (see Figure 7.26):

1. 1% of duplicates in the different databases, except for the Chambers of commerce, where we start with good quality, and set a higher target, i.e., 0.3%.
2. 3% of businesses that do not match in the three databases;
3. 1% inaccuracy of addresses; and

4. an acceptable delay of 3-4 days in the update of information in the three databases.

These targets are a qualitative balance between the "100% quality" ideal (and unreachable) objective, and the present situation. The increased revenues can be estimated assuming that tax fraud decreases proportionally with the number of businesses that can be matched or reached. Other savings will be estimated after having a more precise view of the new ICT infrastructure, provided in the next section.

Quality dimension/ Database matrix	Duplicate objects	Matching objects	Accuracy of names and addressed	Currency
SocialSecurityRegis try	1%	--	99%	3-4 days delay
Accident Insurance Registry	1%	--	99%	3-4 days delay
Chambers of Commerce registry	0.3%	--	99%	2-3 days delay
The three registries together	--	97%	--	--

Fig. 7.26. New quality targets

Choose Improvement Activities

We distinguish between process-driven and data-driven activities. First, we consider process-driven activities. While the present interaction between agencies and businesses involves multiple transactions against the proprietary interfaces of the agencies, a strategic decision of the project is to enable agencies to offer the front office services with a common infrastructure. Such an interface provides a coherent view of the agencies and a single point of access to their business functions. A back-office infrastructure is introduced into the architecture to hide the heterogeneity of the proprietary interfaces as well as their distribution. The approach followed to improve the interaction between administrations is based typically on a *cooperative architecture* that, with some variants, follows the general structure shown in Figure 7.27.

We now provide some comments on the back-office layers. Besides the *connectivity infrastructure*, a *cooperation infrastructure* is shown, including application protocols, repositories, gateways, etc., in which the main goal is to allow each agency to specify and publish a set of cooperative interfaces that include data and application services made available to other agencies. On top of this layer, an *event notification infrastructure* is placed, in which the

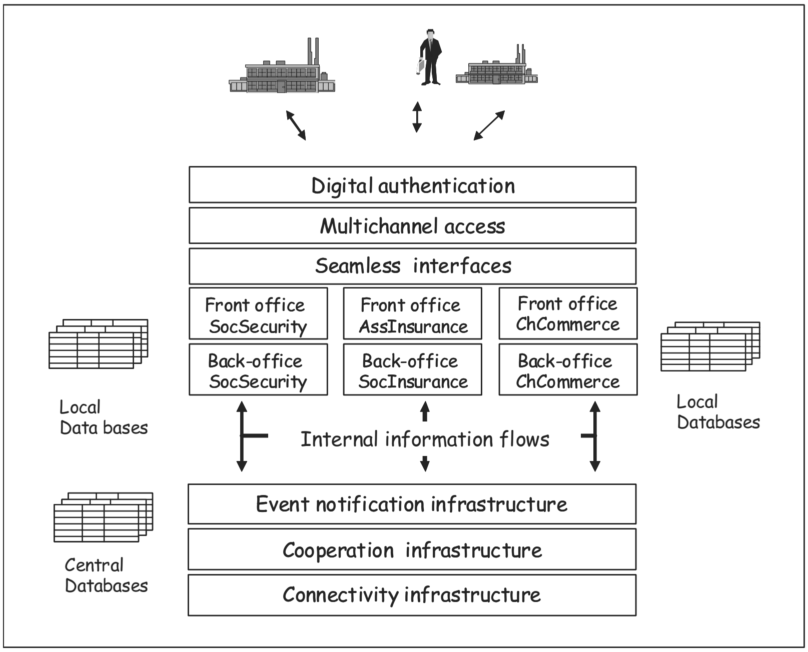


Fig. 7.27. New technological architecture for Government-to-Business interactions

goal is to guarantee synchronization between update events. This layer can be used by an agency when receiving an update from a business. It is published in the cooperative infrastructure; then the information can be subscribed to by all other agencies interested in the update. A number of administrative processes can be reengineered in order to take advantage of this architecture. Specific agencies can be selected as front-end entry points to businesses for specific types of information. In our example, the Chambers of commerce can be involved in updates related to administrative information, while Social Security can manage information related to the workforce, assuming that one of its missions is to collect insurance contributions.

With regard to data-driven activities, in order to make effective reengineered business processes, we need to restructure the data architecture. The two extreme possibilities are

- Create a central database in which all types of managed information on businesses are integrated from the three existing databases.
- Create a light central database in which the records result from the linkage of the identifiers of related business records managed by individual agencies. This new database, which we call *Identifiers database*, is needed to achieve object (business) identification between agencies, and allows for the re-addressing of information in the event notification infrastructure.

The first solution cannot be put into practice because of the autonomy of the agencies. Thus, we choose the second solution. The creation of the Identifiers database requires the object identification activity on the Social security, Accident insurance, and Chamber of commerce registries. At the end of the step, we draw the data/activity matrix (see Figure 7.28). In the databases and data flows we include the new Identifiers database and the new data-flows generated by the event notification infrastructure. We also include the process reengineering activity and the object identification activity discussed above.

Data/Activity	Type of activity	The three databases together	New flows between agencies	The new Identifiers database
Object identification	Data driven	X		
Process Reengineering on update processes	Process driven	X	X	X

Fig. 7.28. The data/activity matrix

We observe that the adoption of the new infrastructure leads to significant savings in costs of interactions. First, we deal with the costs handled by businesses. If businesses reduce interactions by a 3:1 ratio, we estimate that their costs decrease to 70 million Euros a year. With regard to the costs of agencies, in the original system configuration, three front office transactions were required for each business originated update (e.g., change of address), one for each of the three agencies involved in the project; given a cost of 5 Euros for each front office transaction, the total cost is 5 x 3 = 15 Euros. After reengineering, the new update process involves only one front office transaction, plus two new back office transactions to propagate the change. The cost of one back office transaction is 2 Euros, estimated as the sum of fixed costs amortized over the current life of the new system, plus variable costs, considering that initially only one-third of the business events may currently benefit from the new system. Hence, the total cost to the agency goes from 15 Euros to 9 Euros, and can further decrease to a limit cost of 6 Euros as more events are included in the system. Furthermore, if more agencies join the cooperative system, fixed costs will be distributed even further. Finally, provisions can be made to reduce the front office costs, by moving to an entirely paperless and certified submission process for the businesses, with improved up front validation of the input data. This brings the 5 Euros down significantly. Fixing the cost realistically to 6 Euros, we have a decrease in costs from 100 million Euros to 40 million Euros a year.

Choose Techniques for Data Activities

We now have to address the problem of choosing the best techniques for object identification, the main data activity to be performed. Several scenarios can be drawn.

First, we assume that in the past few years partial record linkage activities have been performed between two or all of the three agencies. This is reasonable in the case in which the agencies have a relevant amount of interaction. Consequently, we assume that in previous years they tried to remove, at least partially, errors and misalignments. In this case we have precious knowledge available, consisting of records previously matched and not matched. We take advantage of this knowledge, choosing a probabilistic technique, including a learning activity on frequencies of matching and mismatching.

A second scenario assumes that no previous activity has been performed; but we know that one of the three databases is more accurate than others in certain fields. For instance, one of the agencies is responsible, by law, to certify data related to the names and addresses of businesses. In this case we use the bridging file method.

A third scenario assumes that knowledge is available concerning the behavior of businesses interacting with agencies. For instance, we assume that from data mining tools it has been discovered that specific types of companies, e.g., small family companies, have different part-time activities, changing during different seasons. Consequently, they tend to declare different types of activities to the different agencies, choosing each time the most convenient solutions from an administrative point of view. Among them, certain patterns could be particularly frequent in pairs of records, e.g., <`ice-cream vendor, doorkeeper`>. In this case, it is worthwhile to adopt a knowledge-based technique with a rule based system that includes these types of patterns.

Find Improvement Processes

The analysis performed in previous steps simplifies the identification of improvement processes. We have a unique improvement process (see Figure 7.29) in which we perform the process reengineering activity in parallel, building the publish and subscribe infrastructure and the object identification on the stock. The two activities have to be synchronized at the moment at which the new system becomes operational. Other possibilities, such as data integration, have been excluded in step 8 (choose improvement activities). Note that we do not need a periodic object identification, since business process reengineering, once performed, aligns information hence in the three agencies.

Choose the Optimal Improvement Process

In this case, we have to consider only one improvement process. We have to check for this process to which extent benefits, especially cost savings, exceed the actual cost of quality plus the cost of the project. We apply a simple

Data/Activity	The three DataBases together	New flows between agencies	The new Identifiers DB
Object identification	Perform object identification on the stock and consequent deduplication on the three DBs		
Process Reengineering on update processes	Update first the Chambers ofCommerce DB	Use the P&S Infrastructure toUpdate SocSec DB and SocIns DB	Create the DB and use it in the new interagency update process

Fig. 7.29. An improvement process

methodology where we do not consider issues related to investment analysis and actualization of costs (see [68] and [123]). Concerning actual costs and future cost savings, we have to consider (see the classification provided in Chapter 4) the following major items: (i) costs due to poor data quality, in terms of clerical alignment costs and reduced revenues, and (ii) other costs to businesses and to agencies.

Concerning costs of the data quality improvement project, we have to consider costs related to (i) the object identification activity, in terms of software application and clerical costs (ii) the reengineering of the process, related to set up and maintenance of the publish and subscribe infrastructure.

Reasonable estimates are reported in Figure 7.30. We have estimated some items of the figure in previous sections. With regard to the cost of the improvement project, considering the different subitems, we conclude that the cost of the application architecture is 5 million Euros, and estimate 20% of maintenance costs a year. Object identification is estimated by analogy with previous projects. Finally, increased revenues are estimated on the basis of the percentage of irregular businesses that can be selected with the new target matching values.

In conclusion, if we consider a three-year period, the overall savings and increased revenues come to about 1.2 billion Euros, against a cost of the project that can be considered negligible. If we limit the balance to only data quality related costs and savings, we obtain a net balance of 600 million Euros; the data quality improvement project is extremely worthwhile.

7.6 Summary

Methodologies in general, and, therefore, also DQ methodologies, may be seen as providing common sense reasoning. Their role is to guide in the complex

Costs and benefits	**Once for all**	**Yearly**
Actual costs due to poor data quality		
Clerical alignement costs		10 Ml
Reduced revenues (prudential)		300 Ml
Other costs		
For businesses		200 Ml
For agencies		100 Ml
Costs of the improvement project		
Object identification - automatic	800.000	
Object identification - clerical	200.000	
Application architecture – set up	5Ml	
Application architecture – maintenance		1Ml
Future costs and savings due to improved data quality		
Increased revenues (prudential)		200Ml
Clerical alignement costs		0
Other savings		
For businesses		130Ml
For agencies		60Ml

Fig. 7.30. Costs and savings of the data quality improvement process

decisions to be made, and to understand the knowledge that has to be acquired. At the same time, they have to be adapted to the application domain. A typical error made by designers is to interpret a methodology as an immutable and absolute set of guidelines that have to be applied as they are, without critical examination. The experience gained in working in different domains instructs on how to adapt general guidelines. Furthermore, it is more effective to see the guidelines, phases, tasks, activities, and techniques, which together form a methodology, as a toolbox, where single pieces are to be used in connection and/or in sequence, according to circumstances, and to specific characteristics of the application domain involved in the process.

Another critical issue in DQ methodologies concerns the knowledge available for performing the measurement and improvement defined by the methodology. Sometimes, acquiring the knowledge needed can be very costly, and even impossible. In these cases, the methodology has to be simplified and adapted to knowledge available; otherwise, it is refused by management and users, who are bothered by dozens of questions to which they are not able to reply, and whose purpose they do not understand. As a final point, in order to be effective, methodologies have to be used in connection with automatic tools. Tools and frameworks will be examined in the next chapter.

8 Tools for Data Quality

8.1 Introduction

In previous chapters we have seen that measuring and improving data quality is a complex process, with massive human resource involvement. Techniques discussed in Chapters 4, 5, and 6 are the starting point to automate the activities involved in data quality projects as far as possible. In order to achieve this goal, tools and frameworks have to be developed that encapsulate these techniques.

In this chapter we differentiate between tool, framework, and toolbox. A *tool* is a software procedure that, for one activity or a limited number of activities, e.g., object identification, implements one or a few techniques, e.g., the sorted neighborhood technique. A tool, compared to a technique, is fully automated, and is provided with an interface that allows for the selection of functionalities. A *framework* is a suite of tools that together provide a large amount of DQ functionality for different DQ activities. Thus, the boundary between tools and frameworks is mainly in the scope of the two. A *toolbox* is a tool conceived to compare a set of tools and corresponding techniques, usually for a single DQ activity, including performance and accuracy metrics.

We discuss tools, frameworks, and toolboxes proposed in the literature, limiting our overview to the research world, and not considering the large quantity of commercial tools available for DQ issues. This decision is consistent with the overall research-oriented focus of the book. Readers interested in commercial tools can find comparative reports in the literature providing technical specifications, and some comparisons (see [18] and [94]).

In Section 8.2 we examine tools typically conceived for information systems of a single organization. Initially, we discuss the tools comparatively; then, we detail each of them in a specific subsection. In Section 8.3 we shift our attention to frameworks devised for cooperative information systems. Finally, in Section 8.4 we examine toolboxes specifically conceived to compare tools.

8.2 Tools

Tools considered in this section are introduced comparatively in Figure 8.1; for each tool we quote the main reference, the name, the main activities addressed (according to the classification provided in Chapter 4), the main features, and the application domains in which experiences of use of the tool are reported.

The list of research tools reported in Figure 8.1 is not exhaustive, and provides a picture of the most recent proposals. For instance, the list does not include many government and academic tools for standardization and probabilistic record linkage techniques produced in the 1980s and 1990s, which are described and compared in several papers, e.g., [213] and [88].

Reference	Name	Activities	Features	Application domain
[Raman and Hellerstein 2001]	Potter's wheel	Standardization Object identification and deduplication Data integration – instance-level conflict resolution Profiling (Structure extraction)	Tightly integrates transformations and discrepancy/anomaly detection	Not mentioned
[Caruso et al. 2000]	Telcordia's tool	Standardization Object identification and deduplication	Record linkage tool parametric wrt to distance functions and matching functions	Addresses Tax-payers and their identifiers
[Galhardas et al. 2001]	Ajax	Object identification and deduplication Data integration – instance-level conflict resolution	Declarative language based on logical transformation operators Separates a logical plan and a physical plan	Bibliographic references
[Vassiliadis et al. 2001]	Artkos	Standardization Data integration – instance-level conflict resolution Error localization	Covers all aspects of ETL processes (architecture, activities, quality management) with a unique metamodel	ETL process of an enterprise DW for health appplications and pension data
[Buechi et al. 2003]	Choice Maker	Object identification and deduplication	Uses clues that allows rich expression of the semantics of data	People's names and addresses Business names and addresses Medical data Financial / credit card records
[Low et al. 2001]	Intelliclean	Object identification Choice of representative object	Uses two types of rules, for object identification and object merging	Not mentioned

Fig. 8.1. Names of tools, activities, main features, and application areas

Among the tools, only Telcordia's tool, Ajax [82] and Choice Maker [34] have been engineered into commercial products. The remaining tools are in the state of academic prototypes.

One of the tools, Intelliclean [124], has been described in Chapter 5 as a knowledge-based technique for object identification. In that chapter we described the research paradigms adopted by Intelliclean. Here, we simply compare the tool with the other ones, without further detailing it. With regard to the DQ activities implemented by the tools, the two activities most frequently addressed are (i) object identification, usually coupled with standardization, and (ii) data integration in the form of instance-level conflict resolution. This reflects the central role we have given to the two activities in this book. Notice that Potter's Wheel has a specific activity called *structure extraction*, which is part of the profiling activity mentioned in Chapter 4.

With regard to the main features, Potter's Wheel [166] stresses user friendliness and interactivity in object identification and conflict resolution, resulting in tight integration of transformations and discrepancy/anomaly detection. The main characteristic of Telcordia's tool [43] concerns the high level of flexibility and tailoring in performing record linkage, the tool being parametric with regard to distance and matching functions. The original issues of Ajax are twofold: (i) a declarative language for expressing transformations to be performed on tables for conflict resolution and (ii) the separation of a logical plan for decision for the DQ improvement process and a physical plan for optimizing the choice of techniques. The aim of Artkos [194] is to cover all aspects of the extract, transform, load (ETL) processes typical of data warehouses, namely, architecture, activity, and quality management. A unique descriptive metamodel for this goal is provided. Both Choice Maker and Intelliclean share rules as the main characteristic. We have seen in Chapter 5 that Intelliclean allows expressing domain-dependent rules of two types, (i) duplicate identification rules and (ii) merge/purge rules. Choice Maker provides a wider range of rules, all pertaining to the duplicate identification category, from simple rules such as swaps of groups of fields to complex clues that capture deep properties of the application domain.

With regard to application domains, the most investigated ones are the names and identifiers of individuals/businesses and addresses. Decision support data managed in data warehouses are also a natural area of application of tools, especially in standardization, data integration, instance-level conflict resolution, and error localization issues.

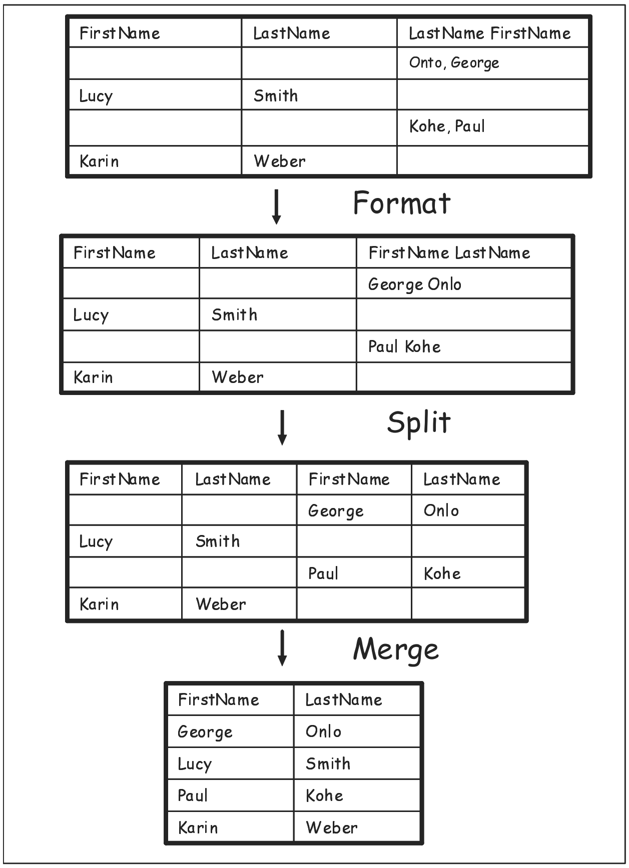
Before concluding the section, it is worth mentioning that there are several tools for profiling, a data activity introduced in Chapter 4, due to increasing interest in the business area. Among the research tools for profiling, we cite Bellman [51] as a good representative of the major profiling tasks, like analysis of the data source content and structure.

8.2.1 Potter's Wheel

Three main data cleaning activities are identified in Potter's Wheel: (i) measuring poor quality to find discrepancies, (ii) choosing transformations to fix the discrepancies, and (iii) applying the transformations on the data.

The major criticisms made by the designers of Potter's Wheel about other tools for data cleaning concern (i) the lack of interactivity and (ii) the significant need for user effort. With reference to the three previously identified data cleaning activities, transformations on data are typically performed with a batch process, operating on a table or the whole database, without any feedback. Furthermore, both discrepancy detection and transformations need significant user effort, making each step of the process difficult and error prone. In the following, we describe in detail how these aspects are addressed by the tool.

Potter's Wheel adopts a small set of transformations that support common transformations used in the DQ improvement process.



FirstName	LastName	LastName FirstName
		Onto, George
Lucy	Smith	
		Kohe, Paul
Karin	Weber	

FirstName	LastName	FirstName LastName
		George Onlo
Lucy	Smith	
		Paul Kohe
Karin	Weber	

FirstName	LastName	FirstName	LastName
		George	Onlo
Lucy	Smith		
		Paul	Kohe
Karin	Weber		

FirstName	LastName
George	Onlo
Lucy	Smith
Paul	Kohe
Karin	Weber

Fig. 8.2. Example of use of transformations in Potter's Wheel

Some of the supported transformations are

1. `Format`, which applies a function to every value in a column. An example of format function is shown in Figure 8.2, where in the last column each <`Lastname, Firstname`> string is transformed into a <`Firstname, Lastname`> one. Format functions can be built-in, or user defined.
2. `Split` (see again Figure 8.2) splits a column into two or more columns, and is used to parse a value into its parts.

3. `Merge` concatenates values in two columns to form a single new column. In Figure 8.2 we see the case of two pairs of columns of <`Firstname, Lastname`> merged into a single pair.

Other transformations help tackle schematic heterogeneities. For instance, `Fold` "flattens" tables by converting one row into multiple rows, merging a set of columns with similar values into one column. In order to reduce the user effort, user can specify the required transformation through examples; the tool produces the function that best matches the provided examples by using algorithms based on the identification of regular expressions. Transformations can be applied interactively, so that their effects can be immediately shown. Furthermore, Potter's Wheel allows the user to undo incorrect transformations. In order to avoid ambiguities, undos are performed logically, by removing the transformation involved from the sequence, and redoing the unchanged transformations.

8.2.2 Telcordia's Tool

Figure 8.3 shows an example of the specification of the record linkage process in Telcordia's tool. Three basic stages can be performed: source selection, standardization (called preprocessing in the tool), and record linkage (called match). Similarly to the activities described in the step "Define improvement process" of CDQM, the methodology described in Chapter 7, the tool allows a DQ administrator to specify complex data analysis flows in which the results of the match between two data sources are used as input to a new matching process with a third data source.

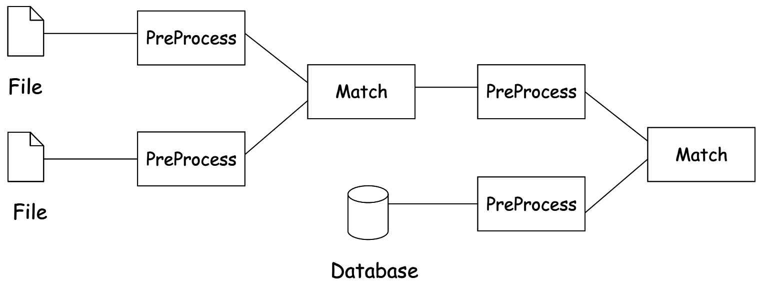


Fig. 8.3. Specification of stages in the Telcordia's Tool

The *source selection* stage enables the selection of the sources to be compared, corresponding to the icons *file* and *database* in Figure 8.3. At this stage, there is an option to select only a representative sample of the database of interest, instead of the entire database, in order to speed up the overall process.

The *preprocessing* stage performs standardization activities of the types described in Chapter 5. Specific examples for the tool include the elimination of special characters, replacement of name aliases, and removal of dashes in dates. Default rules can be specified for particular data types, such as addresses and names; this streamlines the effort involved in selecting rules for repetitive runs on similar data sets. The effects of the activity are shown to the user, who may iterate the activity until he or she is satisfied with the effect the rules have on the data.

The *matching* activity allows the user to select from a variety of matching functions. Examples of available functions include

- records that have an exact or approximate match in specified columns;
- records that match with one column and mismatch with another column; and
- mismatched records based on a measurement such as edit distance.

A claimed advantage of the tool is its ability to create new application specific matching functions that can assist in providing a useful characterization of the causes of data reconciliation problems. This is performed with the following process.

Under the assumption that a number of duplicates has already been detected by the owners of the database, a fraction of all record pairs are supposed to have been correctly labeled as duplicate. Those records can therefore be used as a training set. The strategy then is to determine a set of heuristic classification rules for the record pairs. For several groups of attributes, the frequency of record pairs for which the edit distance (or other comparison function) on such a group lies in a given range is calculated. Such a procedure may proceed with a hierarchical classification process. For instance, the frequency (and the corresponding sets) of record pairs for which the edit distance on the last name lies in a given range is initially calculated, for different ranges. Then, the distribution of members of these sets with respects to additional properties is computed. One can determine, for instance, the fraction of pairs, among those that mismatch slightly on last name, for which the first name disagrees. The resulting taxonomy of groups of pairs can then be used to infer classification rules that are tested and tuned on the training set, and finally applied to the whole set. Note that rules are not generated using the training set; hence, potential problems of overfitting rules are mostly avoided.

8.2.3 Ajax

In this section, we describe the two principal features of Ajax: (i) the declarative language for expressing *transformations* to be performed on tables and (ii) the separation of a *logical plan* and a *physical plan* in the DQ improvement process.

Ajax provides five *transformation operators*. Their composition expresses the principal data transformations proposed in the literature for the object identification activity. The five operators are

1. *Mapping*, is used to split one table into several tables, in order to manage them separately in the DQ process.
2. *View*, which corresponds to a SQL query, augmented by integrity constraints over its result. It can express the same many-to-one mappings of SQL, where each tuple of the output relation results from some combination of tuples taken from the input relation. Different from SQL, integrity constraints can generate exceptions that correspond to specific events in the DQ process (e.g., a field has to be non null).
3. *Matching*, computes an approximate join between two relations where, instead of the equality operator of SQL, a distance function is used to decide which pairs of values are to be joined.
4. *Clustering*, takes a single input relation and returns a single output relation that groups the records of the input relation into a set of clusters. Clusters can be calculated (i) on the basis of the usual *group by* SQL operator or (ii) by means of a distance function.
5. *Merging*, partitions an input relation according to various grouping attributes, and collapses each partition into a single tuple using an arbitrary aggregation function. User-defined aggregation functions can be expressed and used.

A large number of different matching algorithms can be used for implementing the matching operator, the most important of the five operators, depending on the distance function and the approximation adopted. These algorithms adopt techniques such as those described in Chapter 5. Different solutions are provided for the remaining four operators.

The separation of a logical plan and a physical plan corresponds to the typical separation in the design of computer artifacts (such as programs, database schemas, and queries) between a logical phase and a physical phase. We have adopted a similar distinction in CDQM, described in Chapter 7, where we introduced a sequence of decisions for (i) the choice of activities to be performed, (ii) the techniques to be adopted, and (iii) the sequences of the different steps in the process to be followed.

The DQ dimension mainly addressed in Ajax is *accuracy*. The first phase of the DQ improvement, called *logical plan*, concerns the design of the graph of data transformations that are to be applied to the input dirty data. These transformations are conceived in this step without worrying about the specific techniques to be adopted. The focus here is to define *quality heuristics* that can achieve the best accuracy of the results. The second phase, the physical plan, concerns the design of *performance heuristics* that can improve the execution of data transformations without affecting accuracy.

We explain this process of two phases using an example that is inspired by the one discussed in [82] (see Figure 8.4).

Let us suppose we want to perform a deduplication activity (see Figure 8.4a) for a table that represents `Companies`, with attributes `CompanyId`, `Name`, and `TypeofActivity`, and `Owners` of companies, with attributes `SSN`,

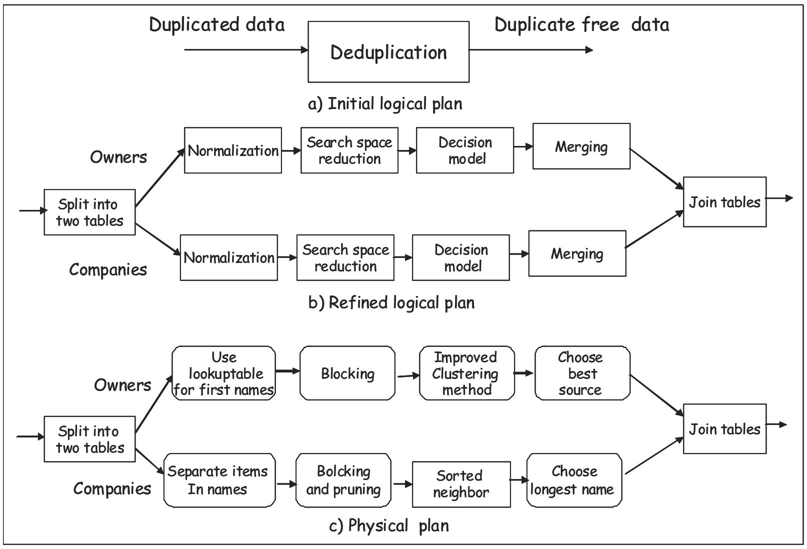


Fig. 8.4. Example of logical plan and physical plan in Ajax

`FirstName`, `LastName`, `DateofBirth`. Due to the different nature of the two types of data, initially (Figure 8.4b) the table can be split into two tables, representing `Owners` and `Companies`. At this point, the typical object identification activities can be performed as described in Chapter 5.

Considering the physical plan (Figure 8.4c), for each of the two flows and for each activity, the most efficient technique can be chosen; for the normalization activity, we can perform for the `FirstName` attribute of `Owners` a comparison with a lookup table; for `Companies`, in which `Name` values are less uniform, we may simply separate items with specific meanings (e.g., Inc.) into different fields. As another example, for the decision step, we may choose an algorithm for `Owners` that customizes and improves a clustering method. For `Companies`, we may choose a sorted neighborhood technique, with window size optimized with respect to the distribution of company names.

8.2.4 Artkos

The main contributions of Artkos, the tool presented in [194], is the presentation of a uniform model covering all the aspects of a data warehouse extract, transform, and load (ETL) process, and of a platform capable of supporting practical ETL scenarios with particular focus on issues of complexity, usability, and maintainability. The most relevant tasks performed in the ETL process include

- identification of relevant information sources;
- extraction of information;

- customization and integration of information coming from sources in a common format;
- cleaning of the resulting tables, on the basis of business rules; and
- propagation of data to the data warehouse.

The designers of Artkos claim that commercial tools are characterized by problems of complexity, usability, and price. To overcome such problems, they ground the architecture of Artkos on a uniform metamodel for ETL processes, covering the principal aspects of data warehouse architecture, activity modeling, and quality management. Artkos includes a metadata repository that has a set of basic assumptions

1. A clear distinction between different layers of instantiation. Therefore, there is (i) a *generic metamodel layer*, which deals abstractly with entities applicable to any data warehouse; (ii) a *metadata layer* dealing with the schemas of a specific data warehouse under examination; and (iii) an *instance layer* representing the real world (as an instance of the previous layers).
2. A clear distinction between perspectives, relying on the separation of (i) the *conceptual perspective*, which represents the world with a model close to the one of the final user; (ii) the *physical perspective*, which covers the data warehouse environment in terms of computer-oriented components; and (iii) the *logical perspective*, which acts as an intermediary between the conceptual and physical layers, though independent of implementation details.

In Figure 8.5 we see the most relevant entities involved in the Artkos metamodel. The generic entities represent three different models, i.e., the process model, the architecture model, and the quality model.

The *process model* describes all the different flows of activities that the designers of the data warehouse decide to perform to implement the ETL process. An *activity* is an atomic unit of work in the chain of data processing. Activities regard components of the architecture model that correspond to input and output tables of one or more databases. An *SQL statement* provides the declarative description of the work performed by each activity. A *scenario* is a set of activities to be executed together. Since data are likely to be affected by quality problems, a large part of the activities of a scenario is dedicated to the elimination of these problems, e.g., the violation of constraints.

Each activity is characterized by an error type and a policy. The *error type* identifies the kind of problem the activity is concerned with. The *policy* expresses how low quality data are to be treated. Several *quality factors* can be defined for each activity, corresponding to dimensions and metrics described in Chapter 2.

Finally, possible error types are listed in Figure 8.5. They give rise to the usual cleaning checks dealt with in the data wharehouse process in the case of

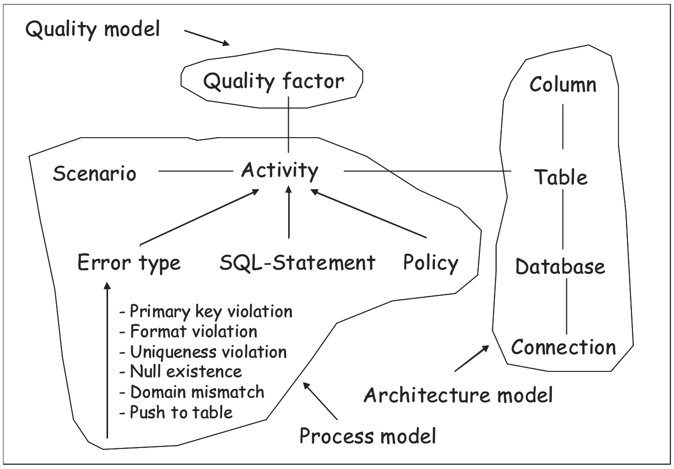


Fig. 8.5. The metamodel of Artkos

relational tables. Such error types can be customized by the user, graphically or declaratively, thus achieving better usability.

8.2.5 Choice Maker

Choice Maker [34] is based on rules, called *clues*. Clues are domain-independent or domain-dependent relevant properties of data. They are used in two phases: offline, the tool determines on a training set the relative importance of the various clues in an attempt to produce for as many examples as possible a decision that is consistent with the human marking, resulting in weight assignments to clues; and at runtime, the trained model is applied to the clues to compute a match probability, that is compared with a given threshold. Several types of clues can be defined in Choice Maker, such as:

1. *swaps of groups of fields*, e.g., swaps of first and last names, such as Ann Sidney with Sidney Ann;
2. *multi-clues*, i.e., groups of clues that differ only by a parameter. For example, one may want to create clues that fire if the first names of records representing persons match and belong to one of five name frequency categories; category 1 contains the very common names (such as "Jim" and "Mike" in the US) and category 5 contains very rare names;
3. *stacked data* describe data that store multiple values for certain fields. For example, current and old addresses may be stored in a relation so that a person can also be located when searching an old address;
4. *complex clues*, that capture a wider set of properties of the application domain.

Complex clues are original types in Choice Maker. They are domain dependent. For an example of a complex clue, assume we have a database of US citizens, a small portion of which is represented in Figure 8.6, and we want to eliminate duplicates from the relation. We can use a decision procedure based on attributes `FirstName`, `LastName`, and `State`. In this case, we probably decide that the pairs of tuples <`1,4`>, <`5,8`> are unmatched, since values of attributes `FirstName` and `LastName` are distant, due probably to several typos. Let us assume that rich senior citizens usually live for one period of the year (around summer) in northern states and for another period (around winter) in southern states. Such a clue can be expressed in Choice Maker as a complex rule, and leads to matched pairs of previously unmatched tuples <`1,4`>, <`5,8`>.

Record #	First Name	Last Name	State	Area	Age	Salary
1	Ann	Albright	Arizona	SW	65	70.000
2	Ann	Allbrit	Florida	SE	25	15.000
3	Ann	Alson	Louisiana	SE	72	70.000
4	Annie	Olbrght	Washington	NW	65	70.000
5	Georg	Allison	Vermont	NE	71	66.000
6	Annie	Albight	Vermont	NE	25	15.000
7	Annie	Allson	Florida	SE	72	70.000
8	George	Alson	Florida	SE	71	66.000

Fig. 8.6. A small portion of the registry of US citizens

The decision procedure can be overridden in special cases. For example, if we trust an identifier such as a social security number, we could use a rule that forces a non-match decision if the two records have different identifier values.

A basic choice for expressing clues is to define a new language, called Clue Maker. This is due to several reasons:

- productivity, because the set of clues written in Clue Maker, is shorter than the same set of clues written in a programming language such as Java;
- usability, because clues are more easily understood by customers;
- correctness, because since the language contains many constructs specific to record matching, it is less error prone than code in Java;
- efficiency, because the language allows for code optimizations that cannot be applied to Java programs due to side effects.

Choice Maker has been used in several projects, with results reported in terms of effort. For instance, a clue set with 200 clues for a complex schema consisting of 60 attributes in ten relations takes two to three person weeks, which is a relatively small amount of time.

8.3 Frameworks for Cooperative Information Systems

We recall the definition of cooperative information system as stated in Chapter 1. A *cooperative information system* (CIS) is a large scale information system that interconnects various systems of different and autonomous organizations, geographically distributed while sharing common objectives. Among the different resources that are shared by organizations, data is fundamental; in real world scenarios, an organization A may not request data from an organization B if it does not "trust" *B's* data, i.e., if A does not know that the quality of the data that B can provide is high. As an example, in an e-Government scenario, in which public administrations cooperate in order to fulfill service requests from citizens and enterprises [21], administrations usually prefer to ask citizens to provide personal data, rather than asking other administrations that own the same data, because the data quality is not known. Therefore, lack of cooperation may occur due to the lack of quality certification. Uncertified quality can also cause a deterioration of the data quality inside organizations.

On the other hand, CISs are characterized by high data replication, i.e., different copies of the same data are stored by different organizations. From a data quality perspective, this is a great opportunity: improvement actions can be carried out on the basis of comparisons between different copies, either to select the most appropriate one or to reconcile available copies, thus producing a new improved version to be sent to all organizations involved.

In this section we consider two frameworks that at the same time address the issues of supporting cooperation and quality improvement in CISs. The two frameworks are introduced in Figure 8.7. With regard to activities addressed, DaQuinCIS [175] covers a large set of issues, in the areas of assessment, object identification, and data integration, and others. The main focus of Fusionplex [135] is the provision of quality-based query processing services that incorporate instance-level conflict resolution. In the rest of the section we discuss two frameworks separately. Other recent frameworks worth to be mentioned for quality based query processing and instance level conflict resolution are iFuice [165] and HumMer [27].

8.3.1 DaQuinCIS Framework

In order to conceive a framework for DQ management in a CIS, the definition of CIS provided in the previous section has to be elaborated, resulting in the following: a *cooperative information system* is formed by a set of organizations $\{ Org_1, \ldots, Org_n \}$ which cooperate through a communication software infrastructure, which provides software services to organizations as well as reliable connectivity. Each organization Org_i is connected to the infrastructure through a gateway G_i, where services offered by Org_i to other organizations are deployed.

This new definition is the basis for the *DaQuinCIS architecture* (see Figure 8.8). The two main components of the architecture are a model for the

Name of Framework	Main references	Activities	Functionalites
DaQuinCIS	[Scannapieco 2004]	Assessment Data correction Object identification Source trustworthiness Data integration: - quality driven query processing - instance conflict resolution	Data quality broker Quality notification Quality factory Rating of sources
Fusionplex	[Motro 2004]	Data integration: - quality driven query processing - instance conflict resolution	Query parser and translator View retriever Fragment factory Query processor Inconsistency detection and resolution Query processor

Fig. 8.7. Comparison between frameworks

organizations to exchange data and quality data, and a set of services that realize data quality functions. The model for data quality proposed is the *data and data quality (D^2Q) model*, described in Chapter 3.

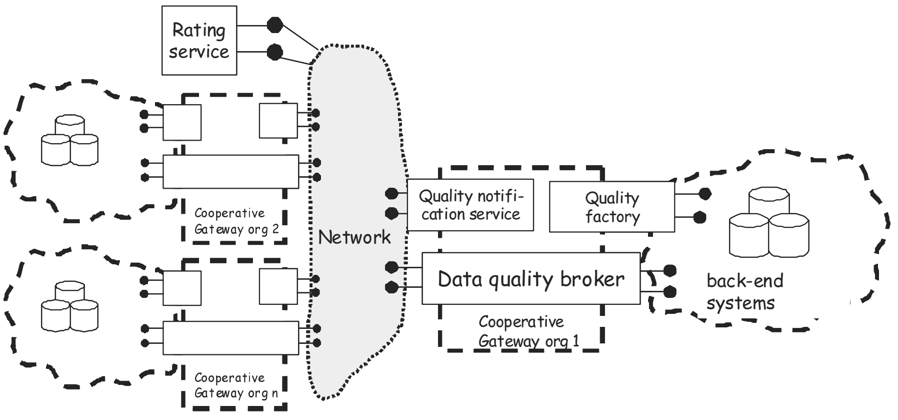

Rating service
Cooperative Gateway org 2
Cooperative Gateway org n
Network
Quality notification service
Quality factory
Data quality broker
Cooperative Gateway org 1
back-end systems

Fig. 8.8. The DaQuinCIS architecture

In the following we focus on services provided by the architecture. The *data quality broker* is the core of the architecture. On behalf of a requesting organization, a query is issued to all organizations, specifying a set of quality requirements on requested data (*quality brokering function*). Different copies of the same data received as responses to the request are reconciled and a best-quality value is selected and proposed to organizations, that can choose to replace their data with higher quality ones (*quality improvement function*). Essentially, the data quality broker is a data integration system [116] which poses quality-enhanced queries over a global schema and selects data satisfying these requirements. The quality-driven query answering process performed by the data quality broker is described in Chapter 6, Section 6.3.2.

The *quality notification service* is a publish/subscribe engine used as a general message bus between architectural components of the different cooperating organizations [174]. It allows quality-based subscriptions for organizations to be notified on changes of the quality of data. For example, an organization may want to be notified if the quality of data it uses degrades below a certain acceptable threshold, or when high quality data are available.

The *quality factory* is responsible for evaluating the quality of internal data of each organization [42]. Its functional modules are shown in Figure 8.9. The quality factory operates as follows. Requests from external users (or the organization information system), are processed by the *quality analyzer*, that performs a static analysis of the values of the data quality dimensions associated with the requested data, and compares them with benchmark quality parameters contained in the *quality repository.*

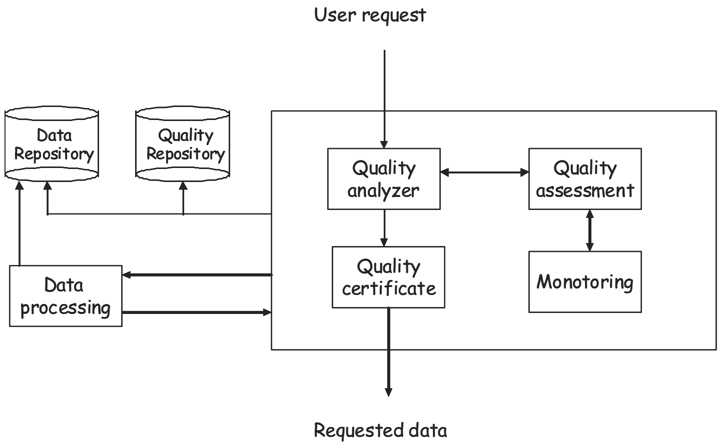


Fig. 8.9. The quality factory

If data values do not satisfy quality requirements, they have to be sent to the *quality assessment* module. This improves the level of data quality, which allows the complete or partial fulfilment of quality requirements. If new values of quality are satisfactory, a quality certificate is associated with the data and is sent to the *data processing* module. This module cooperates with other software applications that are responsible for the final response to the user. A *monitoring* module in charge of monitoring data quality is also included; it executes monitoring operations on the data repository.

The *rating service* associates trust values with each data source in the CIS. They are used to determine the reliability of the quality evaluations made by organizations, which corresponds to its *trustworthiness.* The rating service is a centralized service, to be performed by a third-party organization. The trustworthiness of a source is calculated with reference to a specific data type; therefore, in a cooperative system made up of public administrations, an agency can be more trusted with respect to addresses provided and less trusted with respect to names of citizens. The trustworthiness criterion is computed as a function of several parameters (see [59]), including the number of *complaints* made by other organizations and the number of requests made to each source unit. It may happen that an organization sends a large number of complaints in order to discredit another organization on the same data it owns. In order to prevent such malicious behavior, an adjusting term is introduced into the definition of the trustworthiness criterion.

8.3.2 FusionPlex Framework

The procedures provided by FusionPlex related to quality-driven query processing and instance-level conflict resolution have been described in detail in Chapter 6, Section 6.3.3. Here, we focus on the architecture and functionalities, as shown in Figure 8.10.

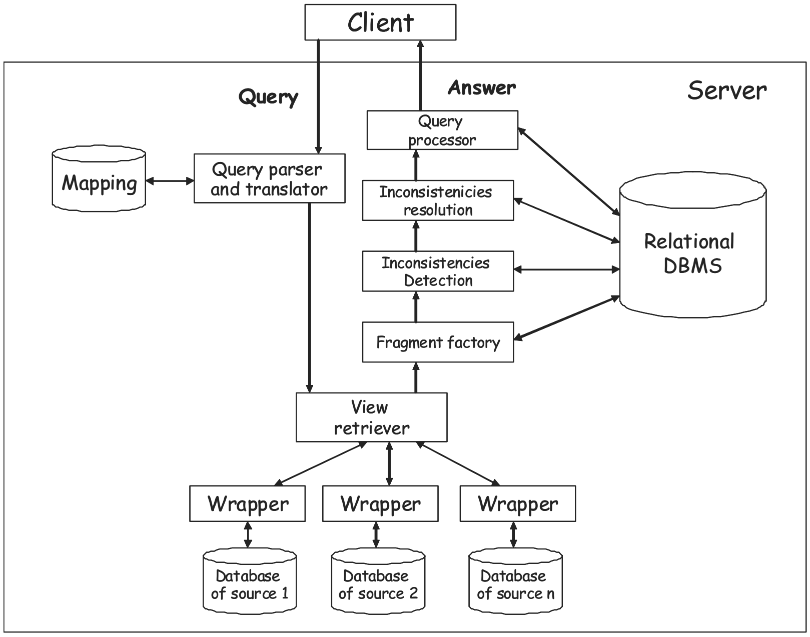


Fig. 8.10. The architecture of Fusionplex

FusionPlex adopts a client-server architecture. The server contains the functionalities that execute the procedure described in Chapter 6:

1. the *query parser and translator* parses the user's query, and determines the source contributions that are relevant to the query;
2. the *view retriever* retrieves the relevant views from the schema mapping;
3. the *fragment factory* constructs the *query fragments*;
4. the *inconsistencies detection module* initially assembles a polyinstance of the answer;
5. the *inconsistencies resolution module* resolves data conflicts in each polytuple according to the appropriate resolution policies;
6. the *query processor* processes the union of all resolved tuples, applies residual aggregations and specified ordering to the query, and returns the query result.

The query parser uses information on the virtual database to be queried, in terms of the global schema, the source schemas, and the mappings among them. Other source characteristics can be added, that quantify a variety of performance parameters of individual sources. Such information is stored at startup time in the *mapping* database, and modified subsequently using a management functionality. Finally, the *relational DBMS* is used to create and manipulate temporary tables.

FusionPlex also provides for powerful and flexible user control over the quality-driven query processing and conflict resolution processes. One user might emphasize the importance of updated information, corresponding to the *timestamp* feature; for another user, *cost* might be most important. User profiles specifying user preferences on the features are managed by the system.

8.4 Toolboxes to Compare Tools

Toolboxes proposed to compare tools focus on the object identification problem. [145] adopts a theoretical approach, while [65] describes a practical tool based on experiments, called Tailor. The two toolboxes are described in the following subsections.

8.4.1 Theoretical Approach

Neiling et al. [145] presents a theoretical framework for comparing techniques. Two aspects are addressed: the complexity of object identification problems and the quality of object identification techniques.

With regard to the first aspect, a reference indicator called *hardness* is introduced. It characterizes the difficulty of an object identification problem; for example, it is intuitive that it is more complex to perform record linkage over two files with low accuracy than over two correct files. As remarked in Chapter 5, the different techniques adopt very specific decision models, characterized in terms of inputs, outputs, and objectives. Therefore, each of the techniques can be more suitable for one class of problems and less suitable for another class of problems. The *hardness* measures how good a technique is for a specific class of problems. The hardness depends on several factors, such as (i) a set of semantic constraints valid in the domain of interest, (ii) the number of pairs to be identified, and (iii) the selectivity of the attribute set that contains identifying information used in the object identification problem.

The second issue addressed in [145] concerns a test framework for the comparison of techniques. The framework consists of a test database, its characteristics (e.g., the existence of semantic keys), several quality criteria for the evaluation of the quality of a solution, and a test specification. The *quality criteria*, inspired by database benchmarks (see [85]), are of two types, respectively *quantitative criteria* and *qualitative criteria*. Quantitative criteria are:

1. *correctness*, the estimation of misclassification rates for test runs;
2. *scalability* with respect to the size of the input;
3. *performance* in terms of computational effort;
4. *cost*, i.e., expenses for the running operations, e.g., hardware and software licenses.

The most important among the above criteria is correctness, which is measured by false negative percentage and false positive percentage, as defined in Chapter 5, Section 5.9.1.

Qualitative criteria include *usability*, *integrability*, *reliability*, *completeness*, *robustness*, *transparency*, *adaptability* and *flexibility*. From these we define three: *usability* is defined as the need for specialized experts and the possibility of automated or incremental updates; *integrability* is considered in the light of existing software architecture functionalities, such as interfaces, data/object exchange, remote control; *transparency* concerns understandability and non-proprietariness of algorithms and results. For definitions of the remaining criteria, see [145].

Similar to the benchmarks available for database management systems, the above set of qualities provides the general criteria for comparing object identification techniques.

8.4.2 Tailor

Tailor [65] is a toolbox for comparing object identification techniques and tools through experiments. The corresponding benchmarking process can be built by tuning a few parameters and plugging in tools that have been developed in-house or are publicly available.

Tailor has four main functionalities (see Figure 8.11), called *layers* in [65], corresponding to (i) the three main record linkage steps discussed in Chapter 5, i.e., *searching method*, *comparison function*, *decision model*, and (ii) *measurement*. Figure 8.11 shows the information flow between the four functionalities, and how the record linkage process operates. The flow is coherent with

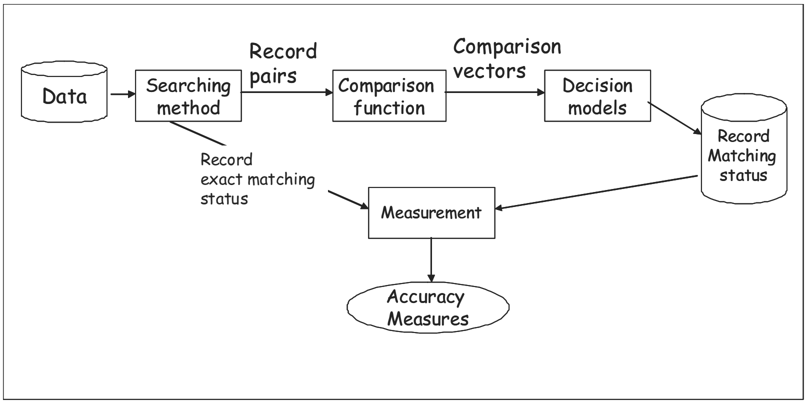


Fig. 8.11. Architecture of Tailor

the general procedure discussed in Chapter 5. At a final stage, a measurement step is performed, to estimate the performance of the decision model.

Layer	Techniques, models and metrics implemented in Taylor
Searching method	Blocking Sorting Hashing Sorted Neighborhood
Comparison function	Hamming distance Edit distance Jaro's algorithm n-grams Soundex code
Decision model	Probabilistic models Fellegy & Sunter familiy Cost based Clustering model Hybrid model

Fig. 8.12. Tailor list of implemented techniques

Figure 8.12 provides a complete list of the various techniques, models and metrics implemented in each of the three record linkage steps. All searching methods and comparison functions mentioned in the figure have been introduced and discussed in Chapter 5. For decision models, the reader may refer to [65] for the clustering model and the hybrid model.

8.5 Summary

Tools and frameworks are crucial for making the techniques and methodologies effective. A comparative analysis of commercial tools is out of the scope of this book. In this chapter we have discussed a specific group of tools and frameworks that closely implement research results. These tools cover various functionalities related to data quality activities, while commercial tools are more focused on specific issues.

In the area of data quality, as in many other areas, there is a temporal gap between research results and their implementation in tools. Furthermore, research groups tend to develop prototypes, characterized by uncertain compatibility and scarce documentation, due to the high investment needed for engineering and selling products. A researcher who aims at using tools in his/her research activity has three choices: (i) use commercial tools, trying to obtain academic licenses, (ii) use public domain tools, extending them with new functionalities, or (iii) develop own tools. The third choice has to be encouraged every time a new technique is conceived in order to experiment and compare results. A theoretical or even qualitative comparison, especially in the data quality area, is seldom possible also when similar paradigms are adopted; only the richness of experimental results can provide evidence of the superiority of a tool with respect to other. Another challenging issue is the production of highly specialized, integrated tools, as an evolution of present tools.

With regard to frameworks, the development process is at an early stage, despite the need for many DQ functionalities in distributed and cooperative information systems. Finally, we notice that the tool is not the solution. In the spirit of this book, this means that the measurement and improvement DQ process has to be carefully planned, using the methodologies discussed in Chapter 7, and the choice of tools has to be addressed only when the relationships between organizations, processes, databases, data flows, external sources, dimensions, and activities to be performed have been deeply understood.

9

Open Problems

In previous chapters we examined all the relevant issues of data quality, from dimensions, to models, activities, techniques, methodologies, tools and frameworks. Among techniques, we have focused mainly on object identification and data integration. We have also emphasized several times the relative immaturity of results and solutions provided in the literature and implemented in tools. In this final chapter we discuss open problems, referring to the more investigated and problematic issues among those addressed above. In Section 9.1 we deal with dimensions and metrics, while in Section 9.2 object identification issues are discussed. Section 9.3 describes data integration, both in trust-aware and in cost-driven query processing. Finally, Section 9.4 considers extensions recently proposed for methodologies. In all sections recent advances are first discussed, followed by an analysis of most relevant open problems.

9.1 Dimensions and Metrics

In Chapter 2, we discussed some data quality dimensions and metrics, showing their meaning and usage by means of examples. However, the problem of defining a reference set of data quality dimensions and metrics is still open. There are several issues to be considered for the purpose:

- Subjective vs. objective assessment. There is no doubt that a database can be of *high quality* for a given application, while being of *low quality* for a different one. Hence the common definition of data quality as "fitness for use". However, such consideration often leads to the wrong assumption that it is not possible to have an objective assessment of quality of data. We claim that for most data quality dimensions (including accuracy, completeness and consistency at least) it makes sense to have objective measures on the basis of which the perceived quality can be evaluated in relation to given user application requirements.

- Domain dependance. For the majority of application domains, a detailed characterization of data quality should take into account the peculiarities of the specific domain. For instance, a set of metrics for evaluating syntactic accuracy must take into account domain dictionaries where available, domain-specific structures (e.g., accuracy of a DNA sequence), etc. This intuition justifies the proposal of domain-specific standards for data quality dimensions and metrics (see Section 2.6). However, in several fields domain specific data quality is being characterized only recently by the scientific community (see e.g. [126] for the biological domain).

Therefore, the following research issues still need investigation. First, a comprehensive set of metrics allowing an objective assessment of the quality of a database should be defined. Metrics should be related to a given data model or format (e.g., relational, XML, or spreadsheets), to a given dimension (typically a single one), and to different degrees of data granularity. Second, appropriate measurement methods are still missing. These methods should allow the clear definition of the sources to measure, e.g., by sampling procedures, the measurement tools, and the measurement precision and errors. Third, there is the need to characterize quality of data in the context of information services. This characterization is required, for instance, as an extension of languages that permit the specification of the quality of service of semantic services (e.g., [219]) or in the definition of service level contracts (e.g., [158]). Fourth, as also highlighted in Chapter 2, data quality dimensions are not orthogonal, instead, we often need to manage tradeoffs among them. Possible tradeoffs within the set of dimensions characterizing quality of data are still worth investigating.

9.2 Object Identification

As described in Chapter 5, traditional record linkage techniques need to be integrated with techniques for matching more complex data structures. We recall that the denomination *object identification* has been used in this book in order to highlight the migration from records to objects, which can be parts of XML documents or pieces of structured information, even stored in different formats, like in *personal information management* (PIM). In Chapter 5 we described the problem of object identification for XML documents; therefore, in Section 9.2.1 we focus on the description of the principal research challenges characterizing XML object identification, not providing further details on the problem specification. In Section 9.2.2 we describe both the problem and the research issues of object identification of information in the PIM context. Finally, in Section 9.2.3 we describe the relationships between record linkage and privacy, which are gaining increasing importance in networked information systems [1].

9.2.1 XML Object Identification

Performing object identification on XML data has two principal peculiarities, when compared to traditional record linkage techniques, namely:

- Identification of the objects to compare. In the relational case, objects coincide with the tuples of a relational table. Conversely, in the XML case, there is the need to identify the XML elements to compare. The delimitation of the boundaries of such elements in the XML document is an issue to be considered. Specifically, the linkage process should be able to identify which portion of the subtree rooted in the elements to compare can be used for performing object identification. Indeed, for deep or wide XML trees, a solution considering the whole subtree can be rather expensive. In Figure 9.1, the problem of identifying which portions of two distinct XML trees referring to a real-world entity "Julia Roberts" is depicted.

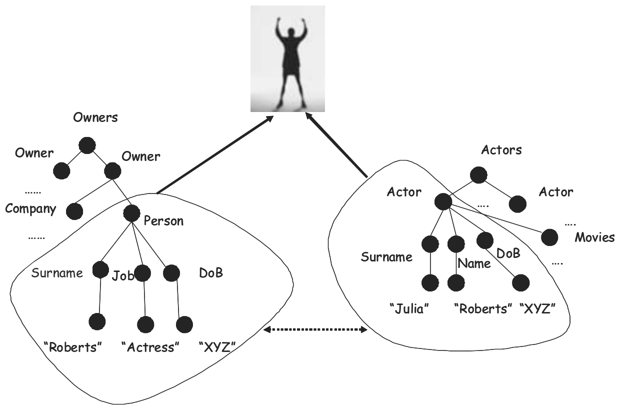


Fig. 9.1. Identifying objects to compare in XML documents

- Exploitation of the flexibility of the XML data model. As a semistructured data model, XML permits the definition of an element in multiple ways, and allows the definition of optional attributes. Such features need to be taken into account in the object identification process in order to make the process as effective as possible. For instance, let us suppose we have two XML documents storing persons. The first document has a schema defining persons, according to a DTD syntax, as follows:

[1] Notice that in this last case we use the term record linkage as we refer again to traditional data structures, like records of a file.

$$<!ELEMENT person1(name, surname)|(surname, DoB)>$$

The second document has a different definition for persons, namely:

$$<!ELEMENT person2(name, surname, DoB)>$$

The matching process should take into account both representations of `person1`. Specifically, instances conforming to the representations of `person1` can be matched with instances conforming to the representation of `person2`, by performing the matching on `name` and `surname` or alternatively on `surname` and `DoB`.

As described in Chapter 5, the above problems have only started to be addressed by recent techniques [207], and therefore there is room for further research investigation.

9.2.2 Object Identification of Personal Information

Personal information management (PIM) has the goal of providing a unified view of information stored on one's desktop. To this end, there is the need to build an integration layer that provides the user with the possibility to store any object of interest according to its semantics, i.e. to relate it to the concepts of a personal ontology, where an object may be a mail, a document, a picture, or any other type of data. In the big picture, such an integration layer could be used for unifying all personal information, even stored on portable devices like PDAs or mobile phones. Focusing on the desktop-level integration, the idea is to enable the user to query the personal ontology, whereas the system carries out the task of suitably processing the query, accessing the different pieces of information involved in the query, and assembling the data into the final answer [108]. This vision can be realized if an object identification activity is performed across a variety of sources including mails, files, pictures, contacts and spreadsheets.

As an example, let us suppose that the same person, e.g., Julia Roberts, is stored as an email contact, as an interviewed person, and as the subject of a picture (see Figure 9.2). In order to build the global object related to Julia Roberts, an object identification activity must be performed, with the purpose of identifying that the three represented objects are actually the same real-world entity.

The described problem gives rise to some open issues, related to the peculiarities of object identification in PIM. First, as remarked in [62], in PIM a small amount of information is used to represent an entity, like for instance a person; indeed, a person representation extracted from an email has only the email address attribute. Conversely, object identification techniques need to rely on several attributes for performing the matching. Second, in a personal information space, information needs to be modeled in a flexible way; hence object identification techniques have problems which are similar to those described for XML object identification.

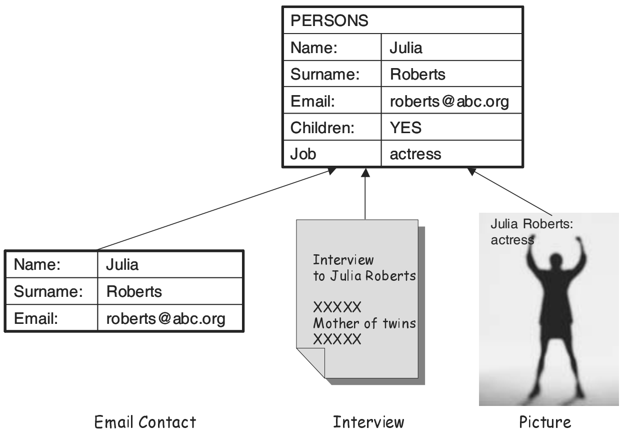


Fig. 9.2. Object identification in PIM

9.2.3 Record Linkage and Privacy

When considering networked information systems, the relationship between record linkage and privacy can be characterized in two ways:

1. Record linkage *prevention* in data publishing. A node can publish its own data so that they are available to the whole networked system. If privacy constraints must be enforced on published data, the node must ensure that no record linkage could be done on published data with the purpose of identifying identities of published individuals and entities.
2. Record linkage *promotion* in data exchanges. Nodes joining a networked system are willing to share information with other nodes. If such information need be privacy protected, record linkage should be enabled while preserving privacy.

In data publishing, a major problem is to assess the risk of privacy violation, once properly disclosed data are published. Typically, anonimyzation does not guarantee zero privacy risk. Indeed, suppose that personal data like `DateOfBirth`, `City` and `MaritalStatus` are published, whereas identifiers like `SSN`, `Name` and `Surname` are removed for the purpose of privacy preservation. By performing record linkage of such data with a public available list, such as an electoral list, it can be easy to identify the individuals with whom the publish data are referenced. Therefore, more sophisticated techniques need to be applied for more properly dealing with privacy assurance.

Among the techniques proposed in the literature, two major classes can be distinguished, namely: perturbation-based techniques and suppression-based techniques. *Perturbation-based techniques* are based on data transformation that include some *noise* for the purpose of privacy preservation; an example of a data perturbation technique consists of swapping data to be published. Data perturbation techniques have been deeply investigated, in the context of statistical databases [2] and privacy-preserving data mining [196]. Some recent proposals for *suppression-based techniques* are briefly described in the following. K-anonymity [171] is a technique that, given a relation T, ensures that each record of T can be indistinctly matched to at least k individuals. It is enforced by considering a subset of T's attributes, called *quasi-identifiers*, and forcing the values that T's records have on quasi-identifiers to appear with at least k-occurrences. In the example above, `DateOfBirth`, `City`, and `MaritalStatus` are examples of quasi-identifiers; for instance, if k=2, in the published data set at least two records must have the same `DateOfBirth`, `City`, and `MaritalStatus` values, and thus are made indistinguishable. A recent technique [112] considers the quantitative evaluation of the privacy risk in case anonymized data are released. In such a work, a database is modeled as a sequence of transactions, and the frequency of an item x in the database is the fraction of transactions that contain that item. A hypothetical attacker can have access to similar data and use them in order to breach the privacy of disclosed data. The knowledge of the attacker is modeled as a belief function that represents the guess that the attacker can make on the actual frequencies of items in the database. In [129], the authors provide an analysis of the *query-view security problem*. Given n views, in such a problem we check if the views disclose any information about a given secret query. The query-view security problem is characterized by means of the notion of *critical tuple* for a query Q that considers a tuple t critical for Q if there are some instances of the database for which dropping t makes a difference. In [129], the authors demonstrate that a query Q is insecure w.r.t. a set of views if and only if they share some common critical tuples. However, so far the problem of characterizing the risk of privacy violation when publishing elementary data in the general case is still open.

In data exchanges, few methods for private record linkage have been proposed. Private record linkage has the purpose of performing record linkage between two sources, say A and B, such that at the end of the process A will know only a set $A \cap B$, consisting of records in A that match with records in B. Similarly B will know only the set $A \cap B$. Of particular importance is the aim that no information will be revealed to A and B concerning records that do not match each other. Figure 9.3 depicts a private record linkage scenario, in which the information that each of the two sources wants to keep secret is represented within a padlock.

Some initial approaches are motivated by the strict privacy requirements of e-health applications [164, 48], or by efficiency issues [5]. Some work that can be related to the problem of private record linkage has also been done in the security area, and involves secure set intersection and secure string comparison. Secure set intersection methods (see [110] for a survey) deal with exact matching and are too expensive to be applied to large databases due to their reliance on cryptography. Furthermore, these protocols deal with the intersection of sets of simple elements, and are not designed for exploiting the semantics behind database records. The problem of securely comparing

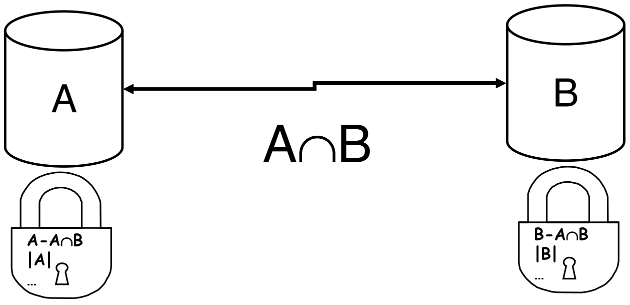


Fig. 9.3. Private record linkage across the two sources A and B

strings has been addressed by homomorphic encryption schemes, characterized by the property that $E(a) * E(b) = E(a + b)$. For example Atallah et al. [10] have proposed an algorithm for comparing sequences based on such schemes. While such an algorithm works for sequence similarity, like DNA sequence comparison, the communication cost of this algorithm, proportional to the product of the sequence lengths, is prohibitive for databases.

9.3 Data Integration

Several data quality problems arise when data integration must be performed. As also remarked in Chapter 6, there are still interesting research issues concerning query processing in data integration systems when assuming that conflicts may arise across sources. Specifically, conflicts can be revealed when tuples from different databases are brought together into an integrated database. From the perspective of the data integration system's semantics, there is no general way to restore consistency, and several possible repairs can be applied. Consistent query answering techniques have been proposed for the purpose of performing such repairs and removing incorrect tuples from the query result. In Chapter 6, we have described some research results and open problems regarding conflict resolution and quality-driven query processing in centralized data integration systems. In Section 9.3.1 we sketch some interesting problems that arise when moving to peer-to-peer (P2P) systems. In Section 9.3.2 we focus instead on the open issues related to query processing in data integration systems when economic models and cost aspects are taken into account.

9.3.1 Trust-Aware Query Processing in P2P Contexts

Peer data management systems (PDMSs) have been proposed as an architecture for decentralized data sharing [188]. Differently from a centralized data integration system, PDMSs do not provide a *mediated schema* on which user queries have to be posed, but data are stored at each peer and only local schemes are used for query answering. P2P systems are characterized by their openness, i.e. a peer can dynamically join or leave the system. When joining a PDMS, a peer has to identify the peers with which to perform data exchanges, and in this respect trust plays an important role. The evaluation of the trust (or confidence) of the data provided by a single peer is crucial because each source can in principle influence the final, integrated result. While several trust and reputation systems have been proposed in the literature (see [106] for a survey), there is still the need to characterize the trust of a peer with respect to provided data and use such information in the query processing step.

More specifically, some open issues involved in trust characterization and evaluation in PDMS include:

- How to model and measure the trust of data provided by a given source. A common distinction is between the reputation of a source, which refers to the source as a whole, and the trust of provided data, e.g., the trust of the mapping that the source establishes with the other sources in a PDMS. Methods for evaluating trust and reputation are needed, with the specific aim of supporting decisions to be taken on result selection.
- Algorithms to compute trust-aware query results. In all the cases a user query can yield a set of different results, such algorithms should allow the selection of the most trustable answer(s). As a secondary effect, whereas a big amount of data is returned as a query result, the trust characterization can provide an "ordering" for processing such results, whereas other priority criteria are not available.

9.3.2 Cost-Driven Query Processing

Information has a cost, which is determined in part by its quality. So, when querying a set of data sources, we have to plan a reasonable cost-quality trade-off. Recently, advances in the technology for large-scale deployment of information services, e.g. over service-oriented software infrastructures, have enabled cost-effective data exchange across organizations. In business terms, this means that it is becoming increasingly feasible for organizations to (i) purchase or otherwise acquire data from other peers, and (ii) exploit their own information assets for marketing purposes. Several studies have analyzed the economic relevance of the potential information market. Public agencies have been found to be the greatest producers of information by far, and the information they create and disseminate is often relevant for both the private and public processes, products and services. In [162] an analysis of the commercial exploitation of *public sector information* is presented both for the USA and the European Union. With the final goal to improve this kind of market in the European Union, rules for managing the reuse of information owned by public sector bodies of the member states have since been issued [71].

The problem of matching information demand to information offer under quality constraints, minimizing on the cost, has been addressed so far in a few papers. The model considered for data sources is a data integration system, as discussed in Chapter 6. A local schema may contain source queries that represent a partial answer to the global query; furthermore, the same source queries may be offered by different suppliers, with different quality levels and costs. In order to satisfy the entire demand, it is necessary to collect source queries from multiple local schemes, possibly selecting among equivalent queries with different quality and cost, in such a way that the quality requirements on the resulting whole are satisfied.

In the approach described in [13], the optimal choice is performed in two steps. First, a query decomposition algorithm selects feasible source queries from the local relations, given a global query. Then, an *integer linear programming* optimization problem is formulated that uses the selected queries and produces a cost-optimal bag of queries that satisfy the entire demand. The composition of qualities of queries is made on the basis of simple composition functions such as, e.g., average.

In the approach of [8], in order to obtain the required data, customers must buy multiple data sets from different providers and then clean and merge them. In this case (see Figure 9.4), a broker architecture intermediates between users and syndicated data providers. On the basis of data quality and cost requirements, the broker builds the most suitable data set by integrating data fragments from different providers. In the selection phase, the broker uses optimization and negotiation mechanisms in order to satisfy requirements. The broker is modeled according to the local-as-view (LAV) perspective, where the data of a provider are represented as views of a global schema, called *broker schema*. The broker is in charge of managing the relationship with providers, and it is also supposed to receive from providers the average value of quality of each data set. Providers have the responsibility to evaluate data quality, along the accuracy, completeness, and timeliness dimensions.

On the basis of data quality and cost requirements, the broker builds the most suitable data set by integrating data from different providers. The optimization approach is based on the tabu search algorithm [83]. A negotiation is started between the broker, behaving on behalf of its user, and the providers, when a solution is infeasible from the point of view of data quality constraints, while still satisfying the constraint referred to price. The aim of the negotiation process is to generate a new set of data fragments that is used in a new exploration; the exploration in performed in an area neighboring to the unfeasible solution, to find the best result for the user query.

Open issues in the area originate from its interdisciplinary background, which includes both technical and economic perspectives. They concern the following aspects:

1. Modeling providers as offering *bundles*, which are indivisible units of data, each one with a single associated price and quality level. Both the cost

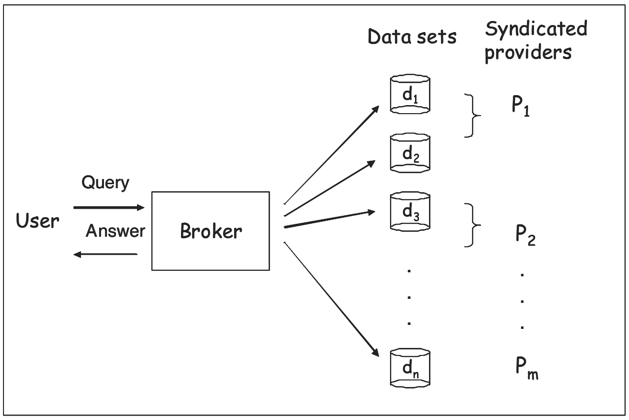


Fig. 9.4. Users, broker, and syndicated data providers

structure behind the production and the selling of digital information goods, and the need to implement anti-competitive strategies can induce more and more data providers to offer indivisible units of different types of data (see for example [191], [14], and [138]).

2. Adopting cost models in which (i) discounts to consumers who acquire two or more (complementary) information goods are provided, or (ii) the cost is a function of the offered quality.
3. Extending economic models to support a *coordinated spot market*, where multiple consumers simultaneously require portions of data with specified quality levels, and multiple suppliers submit their offers and associated quantity-quality matrices to a central public supplier mediator. For instance, the mediator might be in charge of selling data owned by multiple local public agencies to individuals, businesses and other public agencies. In such a case, in order to exploit the quantity/quality discounts as much as possible, the purchasing process could be coordinated by collecting and then matching the overall demand and offer.
4. Considering the case where the improvement in quality of information input to a process has an impact on the quality of information output from the process, resulting in a progressive improvement of the information assets for each participating organization.

9.4 Methodologies

Methodologies for data quality measurement and improvement are evolving in several directions: (i) relating more closely data quality issues and business process issues, and (ii) considering new types of information systems, specifically web information systems.

The relationship between data quality and process quality is a wide area of investigation, due to the relevance and diversity of characteristics of business processes in organizations. We have analyzed the influence of data quality on costs of processes in Chapter 4. Here we broaden the perspective. The different impacts of data quality at the three typical organizational levels, namely operations, the tactical level and the strategic level, are analyzed in [168] reporting interviews and the outcomes of several proprietary studies. Data quality and its relationship with the quality of services, products, business operations, consumer behaviors is investigated in very general terms in [179] and [178], where generic propositions such as "information quality of a firm is positively related to the firm's performance" are stated. The symmetric problem of investigating how improving information production processes positively influences data and information quality is analyzed in [67].

A few papers address more specific issues, and, consequently, present more concrete results. [195] examines the issue of electronic data interchange (EDI), which concerns the exchange of data among organizations using standard formats, and its influence on the efficiency and effectiveness of business processes. EDI may be seen as a DQ technology enabler, since it potentially reduces paper handling related activities, data-entry errors, and data-entry functions. The conclusions of the paper were unforseen. When using EDI technology, the quality of the communication context, namely the degree to which the contexts of the communicating parties are aligned, becomes crucial for the effectiveness of the process. Cases are reported that show a positive effect of EDI on processing time, while poor context DQ resulted in a negative effect. The errors occurring in the EDI process do not occur in the non-EDI one, since in this last case the two parties talk to each other on the phone.

The influence of data quality on extreme conditions in processes, such as disasters, is investigated in [78]. Flaws in accuracy, completeness, consistency, and timeliness are considered in decision making, for example for the 1986 NASA space shuttle Challenger accident, that killed seven people, and the US Navy cruiser Vincennes firing at an Iranian Airbus, that brought 290 people to their death.

The role of information in supply chains is considered in [60], where the *quality robustness* of an information chain is proposed as the ability of the information productive process, of its internal organization in terms of activities and flows among activities, to construct the final information product also in the presence of threats that cause information distortion, transformation variabilities and information failures. A methodological framework called *process quality robustness design* is proposed as a framework for diagnosing, prescribing, and building quality into information chains.

Open problems in the area concern the identification of more precise statistical, probabilistic or functional correlations among data quality and process quality, in the issues related to:

1. more extensive research and empirical validation of the models presented;

2. extension of the analysis to a wider set of dimensions and to specific types of business processes and business areas.

Concerning web information systems, methodologies address several areas: (i) general approaches to web data quality measurement and improvement processes, (ii) more complex data than structured data, namely unstructured data and in particular documents, (iii) new types of dimensions, such as accessibility, considered under different perspectives.

The information quality measurement (IQM) methodology, described in [69] provides general guidelines for measuring and, to a limited extent, improving the quality of information on the web. Considered dimensions are in principle a wide number, ranging from accessibility to consistency, timeliness, currency, comprehensiveness; some of them are evaluated in a narrow sense, e.g., consistency is measured as the number of pages with style guide deviation. IQM consists of two major elements: an action plan on how to conduct the measurement, and an information quality framework that defines which criteria are worth measuring. It is made up of the following phases:

1. measurement planning, made up of: (i) identification of relevant information quality criteria through interviews with stakeholders, (ii) analysis and definition of trade-offs and interdependencies between criteria, (iii) definition of qualitative and quantitative indicators, and (iv) selection of measurement tools for the required indicators;
2. measurement configuration, through weighting of the indicators according to strategic priorities and definition of alert and target values for every indicator;
3. measurement, in terms of (i) data gathering (e.g., monitoring or surveys), (ii) data analysis and presentation;
4. follow-up activities such as corrective measures based on alert indicators, controlling of activities (e.g., assigning responsibilities, and adjustment of measurement according to implementation experiences).

The role played in the process by several tools, such as performance monitoring, site analyzers, traffic analyzers, and web mining is discussed.

Pernici and Scannapieco in [159] propose a model to associate and improve quality information to web data, namely to each item in a web page, a page, and groups of pages, and a methodology for data quality design and management of web information systems. They suggest to enrich methodologies for web information systems design (such as [128] and [102]) with additional steps specifically devoted to data quality design (see Figure 9.5). Several dimensions such as volatility, completability, and semantic and syntactic accuracy as defined in Chapter 2 are covered.

The issue of quality of documents in the web is of increasing relevance, since the number of documents that are managed in web format is constantly

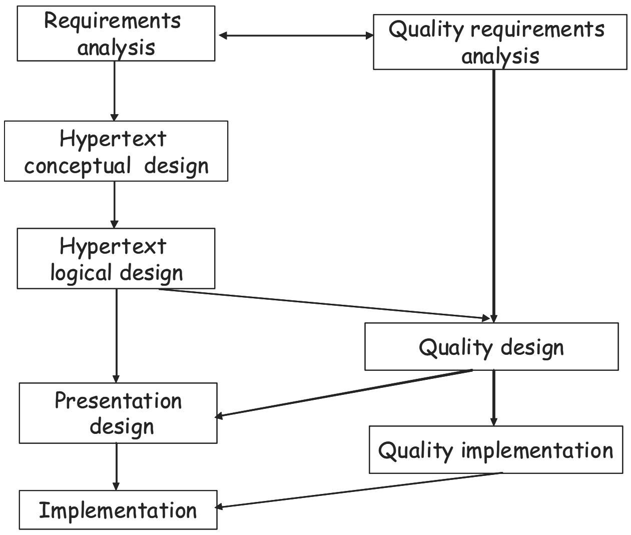


Fig. 9.5. A web information system design methodology (left side) enriched with activities for data quality design (right side)

growing. Several studies have shown that 40% of the material on the net disappears within one year, while a further 40% is altered, leaving only 20% in its original form. Other studies indicate that the average lifetime of a web page is 44 days (see [125]), and the overall web changes completely about four times in a year. As a consequence, the preservation of web data becomes more and more crucial; the term preservation indicates the ability to prevent the losses of information in the web, by storing all significant versions of web documents.

In [41] a methodology to support the preservation process over the entire life cycle of information, from creation, to acquisition, cataloguing, storage, and access is proposed. Main phases are summarized in the following.

1. Each time a new page is published, a procedure named the *static preservation model* has to be executed. At creation time, data are associated with metadata, describing their quality, in terms of accuracy, completeness, consistency, currency and volatility as defined in Chapter 2. Metadata also include properties of the document, such as the author and the document type.
2. Before the acquisition phase is executed, the user specifies acceptable values for all quality dimensions. If new data satisfy quality requirements, they are physically or virtually incorporated into an archive. After acquisition, the web page is cataluoged. If evaluation results do not meet quality requirements, data are returned to their owner with a warning and are not catalogued until their quality is satisfactory. Different suggestions

are provided for data formats to be used in the preservation stage, e.g., translating HTML pages into XML pages.

3. In the publishing stage, when a new page is published by replacing an old web page, the volatility of old data has to be evaluated. If evaluation results indicate that old data are still valid, data are not deleted, but they are associated with a new URL.

A second model called *dynamic preservation model* allows periodic evaluation of the volatility dimension.

Assessment methodologies for evaluating specific qualities of web sites are proposed in [12], [128] and [80]. [12] is specifically focused on *accessibility* as defined in Chapter 2. Accessibility is evaluated on the basis of a mixed quantitative/qualitative assessment. The quantitative assessment activity checks the guidelines mentioned in Chapter 2, provided by the World Wide Web Consortium in [198]. The qualitative assessment is based on experiments performed with disabled users. [80] focuses on the *usability* of the site and proposes an approach based on the adoption of *conceptual logs*, which are web usage logs enriched with meta-data deriving from the application of conceptual specifications expressed by the conceptual schema of the web site. The novelty of the approach is that traditional measures of several qualities of web sites are performed in terms of indicators based on the hypertext representation of the information, e.g., the number of accesses to the different pages, while in this approach new indicators are proposed based on the conceptual representation of the content of the site.

Open problems in the area of methodologies concern:

1. the validation of methodologies. Usually (see references above) a methodology is proposed without any specific experimentation, and with scarce availability of supporting tools. Research on experiments to validate the approaches and on the development of tools to make them feasible is worthwhile;
2. the extension of methodological guidelines to (i) a wider set of dimensions, such as performance, availability, security, accessibility, and to (ii) dependencies among dimensions. For example, a dependency among currency and accuracy is the rule "in 70% of data if an item is not current it is also inaccurate". Knowledge on dependencies, acquired with data mining techniques, can greatly improve the efficiency and effectiveness of the improvement process;
3. in web information systems and in data warehouses, data are managed at different aggregation levels; *quality composition*, as discussed in Chapter 4, should be investigated to derive quality information in aggregate web data from quality information associated with elementary data;
4. the development of more effective assessment methodologies in which, as we have seen in Chapter 7 and in this section when we have discussed [12], both qualitative elements and quantitative indicators are considered.

9.5 Conclusions

In this last chapter we outlined the future development of the data quality research area. In addition to what was presented in this book, in the next ten years there will probably be a widespread increase in contributions in the area, with new paradigms and approaches. Indeed, information is a “plastic” concept and resource, that can hardly be encapsulated into fixed models and techniques. We use textual information to write poetry, facial information to express emotions, musical information to compose or listen to operas. What does it mean that a note in a symphony is executed wrong? It is not easy to formalize this concept, and, probably, it is not useful, since a huge number of phenomena, luckily for us, have to be perceived, and will continue to be perceived, on the basis of our feelings and emotions.

References

1. Abiteboul, S., Buneman, P., and Suciu, D. *Data on the Web: From Relations to Semistructured Data and XML.* Morgan Kaufmann Publishers, 2000.
2. Adam, N. R., and Wortmann, J. C. Security Control Methods for Statistical Databases: A Comparative Study. *ACM Computing Surveys 21*, 4 (1989), 515–556.
3. Agrawal, R., Gupta, A., and Sarawagi, S. Modeling Multidimensional Databases. In *Proc. ICDE 2000* (Birmingham, UK, 1997).
4. Aimetti, P., Missier, P., Scannapieco, M., Bertoletti, M., and Batini, C. Improving Government-to-Business Relationships through Data Reconciliation and Process Re-engineering. In *Advances in Management Information Systems - Information Quality (AMIS-IQ) Monograph*, R. Y. Wang, E. M. Pierce, S. E. Madnick, and C. W. Fisher, Eds. Sharpe, M.E., April 2005.
5. Al-Lawati, A., Lee, D., and McDaniel, P. Blocking-aware Private Record Linkage. In *Proc. IQIS 2005 (SIGMOD Workshop)* (Baltimore, MC, 2005).
6. Amat, G., and Laboisse, B. Une Gestion Operationnelle de la Qualite Donnees. In *Proc. 1st Data and Knowledge Quality Workshop (in conjunction with ECG)* (18th January 2005, Paris, France).
7. Ananthakrishna, R., Chaudhuri, S., and Ganti, V. Eliminating Fuzzy Duplicates in Data Warehouses. In *Proc. VLDB 2002* (Hong Kong, China, 2002).
8. Ardagna, D., Cappiello, C., Comuzzi, M., Francalanci, C., and Pernici, B. A Broker for Selecting and Provisioning High Quality Syndicated Data. In *Proc. 10th International Conference on Information Quality (IQ 2005).*
9. Arenas, M., Bertossi, L. E., and Chomicki, J. Consistent Query Answers in Inconsistent Databases. In *Proc. PODS'99.*
10. Atallah, M. J., Kerschbaum, F., and Du, W. Secure and Private Sequence Comparisons. In *Proc. ACM Workshop on Privacy in the Electronic Society (WPES 2003)* (Washington, Washington DC, 2003).
11. Atzeni, P., and De Antonellis, V. *Relational Database Theory.* The Benjamin/Cummings Publishing Company, 1993.
12. Atzeni, P., Merialdo, P., and Sindoni, G. Web Site Evaluation: Methodology and Case Study. In *Proc. International Workshop on Data Semantics in Web Information Systems (DASWIS 2001)* (Yokohama, Japan, 2001).

13. Avenali, A., Bertolazzi, P., Batini, C., and Missier, P. A Formulation of the Data Quality Optimization Problem in Cooperative Information Systems. In *Proc. CAISE International Workshop on Data and Information Quality* (Riga, Latvia, 2004).
14. Ayres, I., and Nalebuff, B. Going Soft on Microsoft? The EU's Antitrust Case and Remedy. The Economists' Voice.
15. Ballou, D. P., and Pazer, H. L. Modeling Completeness versus Consistency Tradeoffs in Information Decision Contexts. *IEEE Transactions on Knowledge Data Engineering 15*, 1 (2003), 240–243.
16. Ballou, D. P., and Tayi, G. K. Enhancing Data Quality in Data Warehouse Environments. *Communications of the ACM 42*, 1 (1999).
17. Ballou, D. P., Wang, R. Y., Pazer, H., and Tayi, G. K. Modeling Information Manufacturing Systems to Determine Information Product Quality. *Management Science 44*, 4 (1998).
18. Barateiro, J., and Galhardas, H. A Survey of Data Quality Tools. *Datenbank Spectrum 14* (2005), 15–21.
19. Basel Committee on Banking Supervision. http://www.ots.treas.gov.
20. Batini, C., Ceri, S., and Navathe, S. B. *Conceptual Data Base Design: An Entity Relationship Approach*. Benjamin and Cummings, 1992.
21. Batini, C., and Mecella, M. Enabling Italian e-Government Through a Cooperative Architecture. *IEEE Computer 34*, 2 (2001).
22. Batini, C., Nardelli, E., and Tamassia, R. A Layout Algorithm for Data Flow Diagrams. *IEEE Transactions on Software Engineering* (April 1986).
23. Belin, T. R., and Rubin, D. B. A Method for Calibrating False Matches Rates in Record Linkage. *Journal of American Statistical Association 90* (1995), 694–707.
24. Berti-Équille, L. Quality-Adaptive Query Processing over Distributed Sources. In *Proc. 9th Internation Conference on Information Quality (IQ 2004).*
25. Berti-Équille, L. Integration of Biological Data and Quality-driven Source Negotiation. In *Proc. ER 2001* (Yokohama, Japan, 2001).
26. Bertolazzi, P., De Santis, L., and Scannapieco, M. Automatic Record Matching in Cooperative Information Systems. In *Proc. DQCIS 2003 (ICDT Workshop)* (Siena, Italy, 2003).
27. Bilke, A., Bleiholder, J., Böhm, C., Draba, K., Naumann, F., and Weis, M. Automatic Data Fusion with HumMer. In *Proc. VLDB 2005 Demonstration Program* (Trondheim, Norway, 2005).
28. Bitton, D., and DeWitt, D. J. Duplicate Record Elimination in Large Data Files. *ACM Transactions on Databases Systems 8*, 2 (1983).
29. Boag, A., Chamberlin, D., Fernandez, M. F., Florescu, D., Robie, J., and Simèon, J. XQuery 1.0: An XML Query Language. W3C Working Draft. Available from `http:///www.w3.org/TR/xquery`, November 2003.
30. Bouzeghoub, M., and Peralta, V. A Framework for Analysis of Data Freshness. In *Proc. IQIS 2004 (SIGMOD Workshop)* (Paris, France, 2004).
31. Bovee, M., Srivastava, R. P., and Mak, B. R. A Conceptual Framework and Belief-Function Approach to Assessing Overall Information Quality. In *Proc. 6th International Conference on Information Quality (IQ 2001).*
32. Bravo, L., and Bertossi, L. E. Logic Programming for Consistently Querying Data Integration Systems. In *Proc. IJCAI 2003.*

33. Bruni, R., and Sassano, A. Errors Detection and Correction in Large Scale Data Collecting. In *Proc. 4th International Conference on Advances in Intelligent Data Analysis* (Cascais, Portugal, 2001).
34. Buechi, M., Borthwick, A., Winkel, A., and Goldberg, A. ClueMaker: a Language for Approximate Record Matching. In *Proc. 8th International Conference on Information Quality (IQ 2003).*
35. Buneman, P. Semistructured Data. In *Proc. PODS 1997.*
36. Buneman, P., Khanna, S., and Tan, W. C. Why and Where: A Characterization of Data Provenance. In *Proc. International Conference on Database Theory (ICDT 2001)* (London, UK, 2001).
37. Cali, A., Calvanese, D., De Giacomo, G., and Lenzerini, M. On the Role of Integrity Constraints in Data Integration. *IEEE Data Eng. Bull. 25*, 3 (2002), 39–45.
38. Calì, A., Lembo, D., and Rosati, R. On the Decidability and Complexity of Query Answering over Inconsistent and Incomplete Databases. In *Proc. PODS 2003.*
39. Calì, A., Lembo, D., and Rosati, R. Query Rewriting and Answering under Constraints in Data Integration Systems. In *Proc. IJCAI 2003.*
40. Calvanese, D., De Giacomo, G., and Lenzerini, M. Modeling and Querying Semi-Structured Data. *Networking and Information Systems Journal 2*, 2 (1999), 253–273.
41. Cappiello, C., Francalanci, C., and Pernici, B. Preserving Web Sites: a Data Quality Approach. In *Proc. 8th International Conference on Information Quality (IQ 2003).*
42. Cappiello, C., Francalanci, C., Pernici, B., Plebani, P., and Scannapieco, M. Data Quality Assurance in Cooperative Information Systems: a Multi-dimension Certificate. In *Proc. DQCIS 2003 (ICDT Workshop)* (Siena, Italy, 2003).
43. Caruso, F., Cochinwala, M., Ganapathy, U., Lalk, G., and Missier, P. Telcordia's Database Reconciliation and Data Quality Analysis Tool. In *Demonstration at VLDB 2000.*
44. Charnes, A., Cooper, W. W., and Rhodes, E. Measuring the Efficiency of Decision Making Units. *European Journal of Operational Research 2* (1978).
45. Chaudhuri, S., Ganti, V., and Motwani, R. Robust Identification of Fuzzy Duplicates. In *Proc. ICDE 2005.*
46. Chen, Z., Kalashnikov, D. V., and Mehrotra, S. Exploiting Relationships for Object Consolidation. In *Proc. IQIS 2005 (SIGMOD Workshop).*
47. Chiticariu, L., Tan, W., and Vijayvargiya, G. An Annnotation Management System for Relational Databases. In *Proc. VLDB 2004* (Toronto, Canada, 2004).
48. Churches, T., and Christen, P. Some Methods for Blindfolded Record Linkage. *BMC Medical Informatics and Decision Making 4*, 9 (2004).
49. Cui, Y., Widom, J., and Wiener, J. L. Tracing the Lineage of View Data in a Warehousing Environment. *ACM Transactions on Database Systems 25*, 2 (2000), 179–227.
50. Dasu, T., and Johnson, T. *Exploratory Data Mining and Data Cleaning.* J. Wiley Series in Probability and Statistics, 2003.
51. Dasu, T., Johnson, T., Muthukrishnan, S., and Shkapenyuk, V. Mining Database Structure or, How to Build a Data Quality Browser. In *Proc. SIGMOD 2002* (Madison, WI, 2002).

52. Data Warehousing Institute. Data Quality and the Bottom Line: Achieving Business Success through a Commitment to High Quality Data. http://www.dw-institute.com/.
53. Davis, G. B., and Olson, M. H. *Management Information Systems: Conceptual Foundations, Structure, and Development (2nd ed.).* McGraw-Hill, 1984.
54. Davis, R., Strobe, H., and Szolovits, P. What is Knowledge Representation. *AI Magazine 14*, 1 (1993), 17–33.
55. Dayal, U. Query Processing in a Multidatabase System. In *Query Processing in Database Systems*. Springer, 1985, pp. 81–108.
56. De Amicis, F., and Batini, C. A Methodology for Data Quality Assessment on Financial Data. *Studies in Communication Sciences* (2004).
57. De Giacomo, G., Lembo, D., Lenzerini, M., and Rosati, R. Tackling Inconsitencies in Data Integration though Source Preferences. In *Proc. IQIS 2004 (SIGMOD Workshop)* (Paris, France, 2004).
58. De Michelis, G., Dubois, E., Jarke, M., Matthes, F., Mylopoulos, J., Papazoglou, M. P., , Schmidt, J., Woo, C., and Yu, E. Cooperative Information Systems: A Manifesto. In *Cooperative Information Systems: Trends & Directions*, M. Papazoglou and G. Schlageter, Eds. Accademic-Press, 1997.
59. De Santis, L., Scannapieco, M., and Catarci, T. Trusting Data Quality in Cooperative Information Systems. In *Proc. 11th International Conference on Cooperative Information Systems (CoopIS 2003)* (Catania, Italy, 2003).
60. Dedeke, A. Building Quality into Information Supply Chain. In *Advances in Management Information Systems - Information Quality (AMIS-IQ) Monograph*, R. Y. Wang, E. M. Pierce, S. E. Madnick, and C. W. Fisher, Eds. Sharpe, M.E., April 2005.
61. Dempster, A., Laird, N., and Rubin, D. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society Series B 39*, 1 (1977).
62. Dong, X., Halevy, A. Y., and Madhavan, J. Reference Reconciliation in Complex Information Spaces. In *Proc. SIGMOD 2005.*
63. Dublin Core. http://dublincore.org/.
64. Dunn, H. L. Record Linkage. *American Journal of Public Health 36* (1946), 1412–1416.
65. Elfeky, M. G., Verykios, V. S., and Elmagarmid, A. K. Tailor: A Record Linkage Toolbox. In *Proc. 18th International Conference on Data Engineering* (San Jose, CA, 2002).
66. Elmasri, R., and Navathe, S. *Foundamentals of Database Systems (5th ed.).* Addison-Wesley Publishing Company, 1994.
67. English, L. Process Management and Information Quality: How Improving Information Production Processes Improves Information (Product) Quality. In *Proc. 7th International Conference on Information Quality (IQ 2002).*
68. English, L. P. *Improving Data Warehouse and Business Information Quality.* Wiley & Sons, 1999.
69. Eppler, M., and Muenzenmaier, P. Measuring Information Quality in the Web Context: A Survey of State-of-the-Art Instruments and an Application Methodology. In *Proc. 7th International Conference on Information Quality (IQ 2002).*
70. Eppler, M. J., and Helfert, M. A Classification and Analysis of Data Quality Costs. In *Proc. 9th International Conference on Information Quality (IQ 2004).*

71. European Parliament. Directive 2003/98/EC of the European Parliament and of the Council of 17 November 2003 on the Re-use of Public Sector Information. Official Journal of the European Union, 2003.
72. EUROSTAT. http://epp.eurostat.cec.eu.int/pls/portal/.
73. Falorsi, P. D., Pallara, S., Pavone, A., Alessandroni, A., Massella, E., and Scannapieco, M. Improving the Quality of Toponymic Data in the Italian Public Administration. In *Proc. DQCIS 2003 (ICDT Workshop)* (Siena, Italy, 2003).
74. Falorsi, P. D., and Scannapieco, M., Eds. *Principi Guida per la Qualità dei Dati Toponomastici nella Pubblica Amministrazione (in Italian).* ISTAT, serie Contributi, vol. 12. Available at: http://www.istat.it/dati/pubbsci/contributi/Contr_anno2005.htm, 2006.
75. Fan, W., Lu, H., Madnick, S., and Cheungd, D. Discovering and Reconciling Value Conflicts for Numerical Data Integration. *Information Systems 26*, 8 (2001).
76. Fellegi, I. P., and Holt, D. A Systematic Approach to Automatic Edit and Imputation. *Journal of the American Statistical Association 71*, 353 (1976), 17–35.
77. Fellegi, I. P., and Sunter, A. B. A Theory for Record Linkage. *Journal of the American Statistical Association 64* (1969).
78. Fisher, C. W., and Kingma, B. R. Criticality of Data Quality as Exemplified in Two Disasters. *Information Management 39* (2001).
79. Fowler, M. *UML Distilled: A Brief Guide to the Standard Object Modeling Language.* Pearson Education, 2004.
80. Fraternali, P., Lanzi, P. L., Matera, M., and Maurino, A. Model-Driven Web Usage Analysis for the Evaluation of Web Application Quality. *Journal of Web Engineering 3*, 2 (2004), 124–152.
81. Fuxman, A., Fazli, E., and Miller, R. J. ConQuer: Efficient Management of Inconsistent Databases. In *Proc. SIGMOD 2005* (Baltimore, MA, 2005).
82. Galhardas, H., Florescu, D., Shasha, D., Simon, E., and Saita, C. A. Declarative Data Cleaning: Language, Model, and Algorithms. In *Proc. VLDB 2001* (Rome, Italy, 2001).
83. Glover, F., and Laguna, M. *Tabu Search.* Kluver Academic Publishers, 1997.
84. Goerk, M. SAP AG Data Quality@SAP: An Enterprise Wide Approach to Data Quality Goals. In *CAiSE Workshop on Data and Infomation Quality (DIQ 2004)* (Riga, Latvia, 2004).
85. Gray, J. *The Benchmark Handbook for Database and Transaction Systems.* Morgan Kaufmann, 1993.
86. Greco, G., Greco, S., and Zumpano, E. A Logical Framework for Querying and Repairing Inconsistent Databases. *Transactions on Knowledge and Data Engineering 15*, 6 (2003), 1389–1408.
87. Greco, G., and Lembo, D. Data Integration with Preferences Among Sources. In *Proc. ER 2004* (Shangai, China, 2004).
88. Gu, L., Baxter, R., Vickers, D., and Rainsford, C. P. Record Linkage: Current Practice and Future Directions. Technical Report 03/83, CMIS 03/83, Camberra, Australia.
89. Guptil, C., and Morrison, J. *Elements of Spatial Data Quality.* Elsevier Science Ltd, Oxford, UK, 1995.

90. Hall, P. A., and Dowling, G. Approximate String Comparison. *ACM Computing Surveys 12*, 4 (1980), 381–402.
91. Hammer, M., and Champy, J. *Rengineering the Corporation: a Manifesto for Business Revolution*. 2001.
92. Han, J., and Kamber, M. *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2000.
93. Hernandez, M. A., and Stolfo, S. J. Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem. *Journal of Data Mining and Knowledge Discovery 1*, 2 (1998).
94. INFOIMPACT. http://www.infoimpact.com/iqproducts.cfm.
95. International Conference on Information Quality (IQ/ICIQ). http://www.iqconference.org/.
96. International Monetary Fund. http://dsbb.imf.org/.
97. International Organization for Standardization. http://www.iso.org.
98. International Workshop on Data and Information Quality (DIQ). http://www.computing.dcu.ie/research/dataquality/diq/.
99. International Workshop on Information Quality in Information Systems (IQIS). http://iqis.irisa.fr/.
100. International Workshop on Quality of Information Systems (QoIS). http://deptinfo.cnam.fr/qois2006/.
101. INTERPARES Project. http://www.interpares.org.
102. Isakowitz, T., Stohr, E. A., and Balasubramanian, P. RMM: a Methodology for Structured Hypermedia Design. *Communication of the ACM 58*, 8 (1995).
103. Jarke, M., Jeusfeld, M. A., Quix, C., and Vassiliadis, P. Architecture and Quality in Data Warehouses: an Extended Repository Approach. *Information Systems* (1999).
104. Jarke, M., Lenzerini, M., Vassiliou, Y., and Vassiliadis, P., Eds. *Fundamentals of Data Warehouses*. Springer Verlag, 1995.
105. Jaro, M. A. Advances in Record Linkage Methodologies as Applied to Matching the 1985 Cencus of Tampa, Florida. *Journal of American Statistical Society 84*, 406 (1985), 414–420.
106. Josang, A., Ismail, R., and Boyd, C. A Survey of Trust and Reputation Systems for Online Service Provision. *Decision Support Systems* (2005).
107. Kahn, B. K., Strong, D. M., and Wang, R. Y. Information Quality Benchmarks: Product and Service Performance. *Communications of the ACM 45* (2002).
108. Katifori, V., Poggi, A., Scannapieco, M., Catarci, T., and Ioannidis, Y. OntoPIM: how to rely on a Personal Ontology for Personal Information Management. In *Proc. Workshop on The Semantic Desktop 2005*.
109. Kim, W., and Seo, J. Classifying Schematic and Data Heterogeneity in Multidatabase Systems. *IEEE Computer 24*, 12 (1991), 12–18.
110. Kissner, L., and Song, D. Private and Threshold Set-Intersection. Tech. Rep. CMU-CS-05-113, Carnegie Mellon University, February 2005.
111. Krawczyk, H., and Wiszniewski, B. Visual GQM Approach to Quality-driven Development of Electronic Documents. In *Proc. 2nd International Workshop on Web Document Analysis (WDA2003)* (Edinburgh, UK, 2003).
112. Lakshmanan, L. V., Ng, R. T., and Ramesh, G. To Do or Not to Do: the Dilemma of Disclosing Anonymized Data. In *Proc. SIGMOD 2005*.

113. Larsen, M. D., and Rubin, D. B. An Iterative Automated Record Matching using Mixture Models. *Journal of American Statistical Association 79* (1989), 32–41.
114. Lee, Y. W., Strong, D. M., Kahn, B. K., and Wang, R. Y. AIMQ: A Methodology for Information Quality Assessment. *Information and Management* (2001).
115. Lehti, P., and Fankhauser, P. Probabilistic Iterative Duplicate Detection. In *Proc. OTM Conferences 2005.*
116. Lenzerini, M. Data Integration: A Theoretical Perspective. In *Proc. PODS 2002* (Madison, WI, 2002).
117. Levy, A. Y., Mendelzon, A. O., Sagiv, Y., and Srivastava, D. Answering Queries Using Views. In *Proc. PODS 1995* (San Jose, CA, 1995).
118. Lim, E. P., and Chiang, R. H. A Global Object Model for Accommodating Instance Heterogeneities. In *Proc. ER'98* (Singapore, Singapore, 1998).
119. Lin, J., and Mendelzon, A. O. Merging Databases Under Constraints. *International Journal of Cooperative Information Systems 7*, 1 (1998), 55–76.
120. Liu, L., and Chi, L. Evolutionary Data Quality. In *Proc. 7th International Conference on Information Quality (IQ 2002).*
121. Lohningen, H. *Teach Me Data Analysis.* Springer-Verlag, 1999.
122. Long, J. A., and Seko, C. E. A Cyclic-Hierarchical Method for Database Data-Quality Evaluation and Improvement. In *Advances in Management Information Systems - Information Quality (AMIS-IQ) Monograph*, R. Y. Wang, E. M. Pierce, S. E. Madnick, and C. W. Fisher, Eds. Sharpe, M.E., April 2005.
123. Loshin, D. *Enterprise Knowledge Management - The Data Quality Approach.* Morgan Kaufmann Series in Data Management Systems, 2004.
124. Low, W., Lee, M., and Ling, T. A Knowledge-based Approach for Duplicate Elimination in Data Cleaning. *Information Systems 26*, 8 (2001).
125. Lyman, P., and Varian, H. R. How Much Information.
126. Martinez, A., and Hammer, J. Making Quality Count in Biological Data Sources. In *Proc. IQIS 2005 (SIGMOD Workshop).*
127. McKeon, A. Barclays Bank Case Study: Using Artificial Intelligence to Benchmark Organizational Data Flow Quality. In *Proc. 8th International Conference on Information Quality (IQ 2003).*
128. Mecca, G., Merialdo, P., Atzeni, P., and Crescenzi, V. The (Short) ARAENEUS Guide to Web-Site Development. In *Proc. 2nd International Workshop on the Web and Databases (WebDB'99)* (1999).
129. Miklau, G., and Suciu, D. A Formal Analysis of Information Disclosure in Data Exchange. In *Proc. SIGMOD 2004.*
130. Missier, P., and Batini, C. A Multidimensional Model for Information Quality in Cooperative Information Systems. In *Proc. 8th International Conference on Information Quality (IQ 2003).*
131. Missier, P., and Batini, C. A Model for Information Quality Management Framework for Cooperative Information Systems. In *Proc. 11th Italian Symposium on Advanced Database Systems (SEDB 2003)* (Cetraro, Italy, June 2003).
132. Missier, P., and Batini, C. An Information Quality Management Framework for Cooperative Information Systems. In *Proc. International Conference on Information Systems and Engineering (ISE 2003)* (Montreal, Canada, July 2003).

133. Missier, P., Lack, G., Verykios, V., Grillo, F., Lorusso, T., and Angeletti, P. Improving Data Quality in Practice: a Case Study in the Italian Public Administration. *Parallel and Distributed Databases 13*, 2 (2003), 135–160.
134. Monge, A., and Elkan, C. An Efficient Domain Independent Algorithm for Detecting Approximate Duplicate Database Records. In *Proc. SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'97)* (Tucson, AZ, 1997).
135. Motro, A., and Anokhin, P. Fusionplex: Resolution of Data Inconsistencies in the Data Integration of Heterogeneous Information Sources. *Information Fusion* (2005).
136. Motro, A., and Ragov, I. Estimating Quality of Databases. In *Proc. 3rd International Conference on Flexible Query Answering Systems (FQAS'98)* (Roskilde, Denmark, 1998).
137. Muthu, S., Withman, L., and Cheraghi, S. Business Process Reengineering: a Consolidated Methodology. In *Proc. 4th Annual International Conference on Industrial Engineering Theory, Applications and Practice* (San Antonio, TX, 1999).
138. Nalebuff, B. Competing Against bundles. P. Hammond and G. Myles, Eds., Oxford University Press.
139. Naumann, F. *Quality-Driven Query Answering for Integrated Information Systems.* Springer Verlag, LNCS 2261, 2002.
140. Naumann, F., Freytag, J. C., and Leser, U. Completeness of Integrated Information Sources. *Information Systems 29*, 7 (2004), 583–615.
141. Naumann, F., and Häussler, M. Declarative Data Merging with Conflict Resolution. In *Proc. 7th International Conference on Information Quality (IQ 2002).*
142. Naumann, F., Leser, U., and Freytag, J. C. Quality-driven Integration of Heterogenous Information Systems. In *Proc. VLDB'99* (Edinburgh, UK, 1999).
143. Navarro, G. A Guided Tour of Approximate String Matching. *ACM Computing Surveys 31* (2001), 31–88.
144. Nebel, B., and Lakemeyer, G., Eds. *Foundations of Knowledge Representation and Reasoning*, lecture notes in artificial intelligence ed., vol. 810. Springer-Verlag, 1994.
145. Neiling, M., Jurk, S., Lenz, H. J., and Naumann, F. Object Identification Quality. In *Proc. DQCIS 2003 (ICDT Workshop)* (Siena, Italy, 2003).
146. Newcombe, H. B., Kennedy, J. M., Axford, S. J., and James, A. P. F. Automatic Linkage of Vital Records. *Science 130* (1959).
147. Nigam, K., McCallum, A., Thrun, S., and Mitchell, T. Text Classification from Labeled and Unlabeled Documents using EM. *Machine Learning 39* (2000), 103–134.
148. Object Management Group (OMG). Unified Modeling Language Specification, Version 1.5. OMG Document formal/03-03-01, 2003.
149. Office of Management and Budget. Information Quality Guidelines for Ensuring and Maximizing the Quality, Objectivity, Utility, and Integrity of Information Disseminated by Agencies. http://www.whitehouse.gov/omb/fedreg/reproducible.html.
150. OMG. Data Quality and the Bottom Line: Achieving Business Success through a Commitment to High Quality Data. http://www.uml.org/.

151. ORACLE. http://www.oracle.com/solutions/business-intelligence.
152. Ostman, A. The Specifications and Evaluation of Spatial Data Quality. In *Proc. 18th ICA/ACI International Conference* (Stockholm, Sweden, 1997).
153. Ozsu, T., and Valduriez, P. *Principles of Distributed Database Systems.* Prentice Hall, 2000.
154. Papakonstantinou, Y., Abiteboul, S., and Garcia-Molina, H. Object Fusion in Mediator Systems. In *Proc. VLDB 1996* (Bombay, India, 1996).
155. Parssian, A., Sarkar, S., and Jacob, V. Assessing Information Quality for the Composite Relational Operation Join. In *Proc. 7th International Conference on Information Quality (IQ 2002).*
156. Parssian, A., Sarkar, S., and Jacob, V. Assessing Data Quality for Information Products: Impact of Selection, Projection, and Cartesian Product. *Management Science 50*, 7 (2004).
157. Parssian, A., Sarkar, S., and Jacob, V. Assessing Data Quality for Information Products. In *Proc. 20th International Conference on Information Systems (ICIS 99)* (Charlotte, NC, December 1999).
158. Paschke, A., Dietrich, J., and Kulha, K. A Logic Based SLA Management Framework. In *ICSW 2005 Workshop on Semantic Web and Policy Workshop (SWPW 2005)* (2005).
159. Pernici, B., and Scannapieco, M. Data Quality in Web Information Systems. *Journal of Data Semantics* (2003).
160. Pierce, E. M. Extending IP-MAPS: Incorporating the Event-Driven Process Chain Methodology. In *Proc. 7th International Conference on Information Quality (IQ 2002).*
161. Pipino, L. L., Lee, Y. W., and Wang, R. Y. Data Quality Assessment. *Communications of the ACM 45*, 4 (2002).
162. Pira International. Commercial Exploitation of Europe's Public Sector Information, Final Report for the European Commission, Directorate General for the Information Society, October 2000.
163. Poirier, C. A Functional Evaluation of Edit and Imputation Tools. In *UN/ECE Work Statistical Data Editing* (Rome, Italy, 2-4 June 1999).
164. Quantin, C., Bouzelat, H., Allaert, F., Benhamiche, A., Faivre, J., and Dusserre, L. How to Ensure Data Security of an Epidemiological Follow-up: Quality Assessment of an Anonymous Record Linkage Procedure. *International Journal of Medical Informatics 49*, 1 (1998).
165. Rahm, E., Thor, A., Aumueller, D., Do, H. H., Golovin, N., and Kirsten, T. iFuice - Information Fusion Utilizing Instance Correspondences and Peer Mappings. In *Proc. 8th International Workshop on the Web and Databases (WebDB 2005)* (2005).
166. Raman, V., and Hellerstein, J. M. Potter's Wheel: An Interactive Data Cleaning System. In *Proc. VLDB 2001* (Rome, Italy, 2001).
167. Redman, T. C. *Data Quality for the Information Age.* Artech House, 1996.
168. Redman, T. C. The Impact of Poor Data Quality on the Typical Enterprise. *Communications of the ACM* (1998).
169. Redman, T. C. *Data Quality The Field Guide.* The Digital Press, 2001.
170. Saaty, T. L. *The Analytic Hierarchy Process.* McGraw-Hill, 1980.
171. Samarati, P. Protecting Respondents' Identities in Microdata Release. *IEEE Transactions on Knowledge and Data Engineering 13*, 6 (2001), 1010–1027.
172. Sarawagi, S., and Bhamidipaty, A., Eds. *Interactive Deduplication Using Active Learning* (Edmonton, Alberta, Canada, 2002).

173. Scannapieco, M., and Batini, C. Completeness in the Relational Model: A Comprehensive Framework. In *Proc. 9th International Conference on Information Quality (IQ 2004)*.
174. Scannapieco, M., Pernici, B., and Pierce, E. M. IP-UML: A Methodology for Quality Improvement based on IP-MAP and UML. In *Advances in Management Information Systems - Information Quality (AMIS-IQ) Monograph*, R. Y. Wang, E. M. Pierce, S. E. Madnick, and C. W. Fisher, Eds. Sharpe, M.E., April 2005.
175. Scannapieco, M., Virgillito, A., Marchetti, C., Mecella, M., and Baldoni, R. The DaQuinCIS Architecture: a Platform for Exchanging and Improving Data Quality in Cooperative Information Systems. *Information Systems 29*, 7 (2004), 551–582.
176. Schallehn, E., Sattler, K. U., and Saake, G. Extensible and Similarity-Based Grouping for Data Integration. In *Proc. of the ICDE 2002* (San Jose, CA, 2002).
177. Shankaranarayan, G., Wang, R. Y., and Ziad, M. Modeling the Manufacture of an Information Product with IP-MAP. In *Proc. 5th International Conference on Information Quality (IQ 2000)*.
178. Sheng, Y. H. Exploring the Mediating and Moderating Effects of Information Quality on Firms? Endeavor on Information Systems. In *Proc. 8th Internationa Conference on Information Quality (IQ 2003)*.
179. Sheng, Y. H., and Mykytyn Jr., P. P. Information Technology Investment and Firm Performance: A Perspective of Data Quality. In *Proc. 7th Internationa Conference on Information Quality (IQ 2002)*.
180. Smith, T. F., and Waterman, M. S. Identification of Common Molecular Subsequences. *Molecular Biology 147* (1981), 195–197.
181. Stoica, M., Chawat, N., and Shin, N. An Investigation of the Methodologies of Business Process Reengineering. In *Proc. of Information Systems Education Conference* (2003).
182. Stolfo, S. J., and Hernandez, M. A. The Merge/Purge Problem for Large Databases. In *Proc. SIGMOD 1995* (San Jose, CA, 1995).
183. Storey, V., and Wang, R. Y. Extending the ER Model to Represent Data Quality Requirements. In *Data Quality*, R. Wang, M. Ziad, and W. Lee, Eds. Kluver Academic Publishers, 2001.
184. Storey, V. C., and Wang, R. Y. An Analysis of Quality Requirements in Database Design. In *Proc. 4th International Conference on Information Quality (IQ 1998)*.
185. Su, Y., and Jin, Z. A Methodology for Information Quality Assessment in the Designing and Manufacturing Processes in Mechanical Products. In *Proc. 9th International Conference on Information Quality (ICIQ 2004)*.
186. Tamassia, R., Batini, C., and Di Battista, G. Automatic Graph Drawing and Readability of Diagrams. *IEEE Transactions on Systems, Men and Cybernetics* (1987).
187. Tarjan, R. E. Efficiency of A Good But Not Linear Set Union Algorithm. *Journal of the ACM 22*, 2 (1975), 215–225.
188. Tatarinov, I., and Halevy, A. Y. Efficient Query Reformulation in Peer-Data Management Systems. In *Proc. SIGMOD 2004*.
189. Tejada, S., Knoblock, C. A., and Minton, S. Learning Object Identication Rules for Information Integration. *Information Systems 26*, 8 (2001).

190. Ullman, J. D. *Principles of Database and Knowledge-Base Systems.* Computer Science Press, 1988.
191. Ulusoy, G., and Karabulut, K. Determination of the Bundle Price for Digital Information Goods. University of Sabanci, Istanbul, 2003.
192. U.S. National Institute of Health (NIH). http://www.pubmedcentral.nih.gov/.
193. van der Aalst, W., and ter Hofstede, A. YAWL: Yet Another Workflow Language. *Information Systems 30*, 4 (2005), 245–275.
194. Vassiliadis, P., Vagena, Z., Skiadopoulos, S., Karayannidis, N., and Sellis, T. ARTKOS: Toward the Modeling, Design, Control and Execution of ETL Processes. *Information Systems 26* (2001), 537–561.
195. Vermeer, B. H. P. J. How Important is Data Quality for Evaluating the Impact of EDI on Global Supply Chains ? In *Proc. HICSS 2000.*
196. Verykios, V. S., Elmagarmid, A. K., Bertino, E., Saygin, Y., and Dasseni, E. Association Rule Hiding. *IEEE Transaction on Knowledge and Data Engineering 16*, 4 (2004).
197. Verykios, V. S., Moustakides, G. V., and Elfeky, M. G. A Bayesian Decision Model for Cost Otimal Record Matching. *The VLDB Journal 12* (2003), 28–40.
198. W3C. http://www.w3.org/WAI/.
199. Wand, Y., and Wang, R. Y. Anchoring Data Quality Dimensions in Ontological Foundations. *Communications of the ACM 39*, 11 (1996).
200. Wang, R. Y., Chettayar, K., Dravis, F., Funk, J., Katz-Haas, R., Lee, C., Lee, Y., Xian, X., and S., B. Exemplifying Business Oppurtunities for Improving Data Quality from Corporate Household Research. In *Advances in Management Information Systems - Information Quality (AMIS-IQ) Monograph*, R. Y. Wang, E. M. Pierce, S. E. Madnick, and C. W. Fisher, Eds. Sharpe, M.E., April 2005.
201. Wang, R. Y., Lee, Y. L., Pipino, L., and Strong, D. M. Manage Your Information as a Product. *Sloan Management Review 39*, 4 (1998), 95–105.
202. Wang, R. Y., and Madnick, S. E. A Polygen Model for Heterogeneous Database Systems: The Source Tagging Perspective. In *Proc. VLDB'90* (Brisbane, Queensland, Australia, 1990), pp. 519–538.
203. Wang, R. Y., Pierce, E., Madnick, S., and Fisher, C. *Information Quality, Advances in Management Information Systems.* M.E. Sharpe, Vladimir Zwass Series, 2005.
204. Wang, R. Y., Storey, V. C., and Firth, C. P. A Framework for Analysis of Data Quality Research. *IEEE Transaction on Knowledge and Data Engineering 7*, 4 (1995).
205. Wang, R. Y., and Strong, D. M. Beyond Accuracy: What Data Quality Means to Data Consumers. *Journal of Management Information Systems 12*, 4 (1996).
206. Wang, R. Y., Ziad, M., and Lee, Y. W. *Data Quality.* Kluwer Academic Publisher, 2001.
207. Weis, M., and Naumann, F. DogmatiX Tracks down Duplicates in XML. In *Proc. SIGMOD 2005.*
208. White, C. Data Integration: Using ETL, EAI, and EII Tools to Create an Integrated Enterprise. http://ibm.ascential.com, 2005.
209. Wiederhold, G. Mediators in the Architecture of Future Information Systems. *IEEE Computer 25*, 3 (1992).

210. Winkler, W. Improved Decision Rules in the Fellegi-Sunter Model of Record Linkage. In *Proc. of the Section on Survey Research Methods, American Statistical Association* (1993).
211. Winkler, W. E. Using the EM Algorithm for Weight Computation in the Fellegi and Sunter Modelo of Record Linkage. In *Proc. of the Section on Survey Research Methods, American Statistical Association* (1988).
212. Winkler, W. E. Matching and Record Linkage. In *Business Survey Methods.* Wiley & Sons, 1995.
213. Winkler, W. E. Matching and Record Linkage. In *Business Survey Methods.* Wiley & Sons, 1995.
214. Winkler, W. E. Machine Learning, Information Retrieval and Record Linkage. In *Proc. of the Section on Survey Research Methods, American Statistical Association* (2000).
215. Winkler, W. E. Methods for Evaluating and Creating Data Quality. *Information Systems 29*, 7 (2004).
216. Winkler, W. E. Quality of Very Large Databases. Technical Report RR-2001/04, U.S. Bureau of the Census, Statistical Research Division, Washington, Washington DC, 2001.
217. Wiszniewski, B., and Krawczyk, H. Digital Document Life Cycle Development. In *Proc. 1st International Symposium on Information and Communication Technologies (ISICT 2003)* (Dublin, Ireland, 2003).
218. Yan, L. L., and Ozsu, T. Conflict Tolerant Queries in AURORA. In *Proc. CoopIS'99* (Edinburgh, UK, 1999).
219. Zhou, C., Chia, L. T., and Lee, B. S. QoS Measurement Issues with DAML-QoS Ontology. In *Proc. 2005 IEEE International Conference on e-Business Engineering (ICEBE'05)* (2005).

Index