A MULTIDIMENSIONAL MODEL FOR INFORMATION QUALITY IN COOPERATIVE INFORMATION SYSTEMS

(research-in-progress)

Paolo Missier
Universita' di Milano Bicocca, Milano, Italy
pmissier@disco.unimib.it

Carlo Batini
Universita' di Milano Bicocca, Milano, Italy
batini@disco.unimib.it

Abstract. Cooperative Information Systems implement cross-organization processes by exchanging information among each other in a mutual supplier--customer relationship. With respect to the quality of the information received, each customer system has requirements that must be met by its supplier systems. In this paper, we model the quality profile associated to a supplier as a multidimensional data cube, show how requirements can be formally expressed by referring to views over the cube, and thus provide a precise notion of adequacy of the supplier with respect to its customers' requirements, and a way for customers to rank their suppliers with respect to the quality profile they offer.

Key Words: Information Quality, Cooperative Information Systems, Multidimensional Modeling

INTRODUCTION

Cooperative Information Systems (CIS for short), defined for instance in [1], manage the internal business and system processes of a single organization, but they also take part in inter-organization, cooperative processes. We call CIS federation a collection of such cooperative systems[1]. To frame the problem of managing Information Quality in CIS, we start from the metaphor first adopted by the TDQM quality control methodology [27][28][3], wherein information is viewed as a product (Information Product, or IP) that is supplied by a CIS and used by other CISs, during the execution of cooperative processes. Combined with the use of IP-MAP graphical notation for production processes [26], TDQM provides a useful process-centric view of information quality. At a very high level, TDQM consists of four main steps, namely identifying those IPs whose quality is critical according to the needs of its users, and defining their Quality requirements (Define IP); identifying effective quality metrics to measure the quality of those IPs (Measure IP); performing data-driven and process-driven analysis to uncover the causes for poor quality on those IPs (Analyze IP); and finally, devising and implementing strategies to improve IQ by acting on those causing factors (Improve IP). These steps are repeated in a loop, following the classic Do-Plan-Check-Act cycle [7].

Information quality improvement methodologies like TDQM do provide an adequate framework for the management of IQ programs. However, they do not take the specific features of cooperative information systems into account. In our work, we carry the IP metaphor further. In a CIS federation, an organization

[1] This definition departs slightly from the notion of CIS given in [8], where CIS indicates the federation itself.

is modeled as a collection of processes that transform input information flows into output information flows that carry a stream of IPs. Thus, IPs are exchanged by organizations through flows, in the context of specific cooperative processes. Similar to manufactured items, IPs that are produced by processes managed by one organization, may be acquired by other processes and used to produce other IPs. Following traditional manufacturing practice, on the IP producer side we may characterize the quality of the individual items produced, and by extension, we may associate a *quality profile* to a whole organization of producer processes. Such profile represents the quality that the organization is willing to *offer* to its customers, i.e. to other organizations that require that information for use within a cooperative process. Symmetrically, on the IP customer side we can define the notion of *quality demand*, to express acceptable quality levels for the information items those customers are going to acquire. Ultimately, we can frame the problem of managing information quality within an organization, as the problem of matching the quality profile offered by that organization to the quality requested by the organization's customers.

The main contribution of this paper is a formal framework for expressing quality offer and demand in the CIS context. The framework models both the structure of a cooperative organization (Data Model) and its quality profiles (Quality Model) in a uniform, hierarchical way. We start by associating quality profiles to the elementary information items that the organization produces and consumes during the execution of cooperative processes, and then move up the hierarchy until a summary quality profile is associated to the organization itself. To achieve this, quality profiles are modeled as multidimensional data cubes. The cube dimensions reflect the hierarchical structure of the Data model, while its measure carries quality information about the information items. Using slicing and roll-up operations on the cube, various quality views are generated. These views are used to focus a general supplier profile to specific items and levels of granularity of interest to each customer, within the scope of a cooperative process. Furthermore, we define the notion of *distance* between quality profiles, that enables organizations to negotiate quality levels based on offer and demand. Finally, we define the *adequacy* of a supplier organization as the distance of a specific view over its quality offer, from the customer's quality demand.

Some of the elements in the framework are generic. In order to provide practical support to IQ management, the framework must be instantiated, by providing organization-specific definitions for its abstract elements, including the following:

- a set of quality dimensions and a definition of quality descriptors for those dimensions;
- a set of aggregating functions defined over quality values;
- scoring functions to be used by customers to rank suppliers' profiles.

To illustrate the framework, we provide sample definitions for some of these abstract elements. We are currently working on the implementation of a prototype to demonstrate the feasibility of the approach, and on its application to a real case study.

The paper is organized as follows. After concluding this section with related work, we introduce the basic definitions for our quality model in Section 2, and then formalize the notion of Quality Cube of Quality Demand and Offer, in Section 3. A note on further work concludes the paper. Space limitations prevent us from presenting a complete illustrative example. Our in-progress case study can be found as a deliverable of the DaQuincis project [6].

Related Work

Extensive literature exists on Information Quality Management, which has been investigated over the years both from the strictly technical perspective of data cleaning, and as a business process management

problem. In the technical arena, research work has concentrated on techniques for data cleaning. Verykios et al. [29] offer a survey of recent literature on data cleaning and record matching. The specific problem of matching similar records in different datasets, known as record linkage problem in Newcombe [19], was later addressed in Fellegi and Sunter [10] using statistical models and algorithms. Record matching algorithms that perform well over large datasets have been presented for instance in [15].
As the field matured, toolkits for dealing with dirty data over large and multiple data sources started to emerge. Galhardas et al. [13] propose a framework for data reconciliation that includes operators for data transformation, duplicate elimination and multi-table matching. In the Telcordia data quality analysis toolkit [1], data analysts specify complex data cleaning workflows by linking together basic data processing blocks.

The term *fitness for use* has been proposed in [14] to denote the extent to which a product successfully serves the purpose of consumers. By extension, Information Quality requirements [15] capture the fitness for use for Information Products, taking the view that the quality of information is always defined relative to its intended use, rather than a priori.

A number of comprehensive methodologies are meeting considerable success in business consulting. The case study illustrated in [17] for the Italian Public Administration is an example of successful application of a general product-view approach, that has been adapted for a specialized application domain.

Principles of Total Quality Data Management are presented in [9] and applied to the business environment and specifically to manage Data Warehousing Information quality. Other approaches to process management that impacts information quality include SixSigma [22][19] and Quality Function Deployment, or QFD [22]. The latter focuses on the translation of subjective and informal user requirements into specific process and technical requirements, in such a way that the information production processes are fully specified.

Alongside these methodologies and toolkits, a new breed of information quality management approaches is emerging that is specific to the area of CIS. In a cooperative environment, relevant data quality issues include the assessment of the quality of the data owned by each organization; methods and techniques for exchanging quality information, and for improving the information quality within each cooperating organization; and the differences in the semantics of the data that is managed by different organizations.
Several solutions have recently been proposed to deal with these issues. Models for quality metadata description are described in [23][25][28]. CIS-oriented frameworks are presented in [10] and [3].

THE QUALITY MODEL: BASIC DEFINITIONS

We begin by modeling the notion of information items that are exchanged by processes in the context of a cooperative workflow. We then extend the model to include the notion of quality profile associated to information supplier organizations.

A Data Model for Information Items exchange

The data model diagram in Figure 1 shows the main entities and their relationships. The model describes the I/O behavior of usiness Processes that are managed by Organizations (left side of the diagram). Note that we are not interested in modeling the internal behavior of processes. Processes exchange information among hemselves, through Information Flows that carry Information Items from a Supplier Process to a Consumer Process. Thus, a process *P* transforms input flows into output flows. We also consider internal flows that account for information transformations performed by a process on its local data. We denote

the set of input and output flows for a process *P* as *in(P)* and *out(P)*, respectively, and the set of processes managed by organization *o* as *proc(o)*.

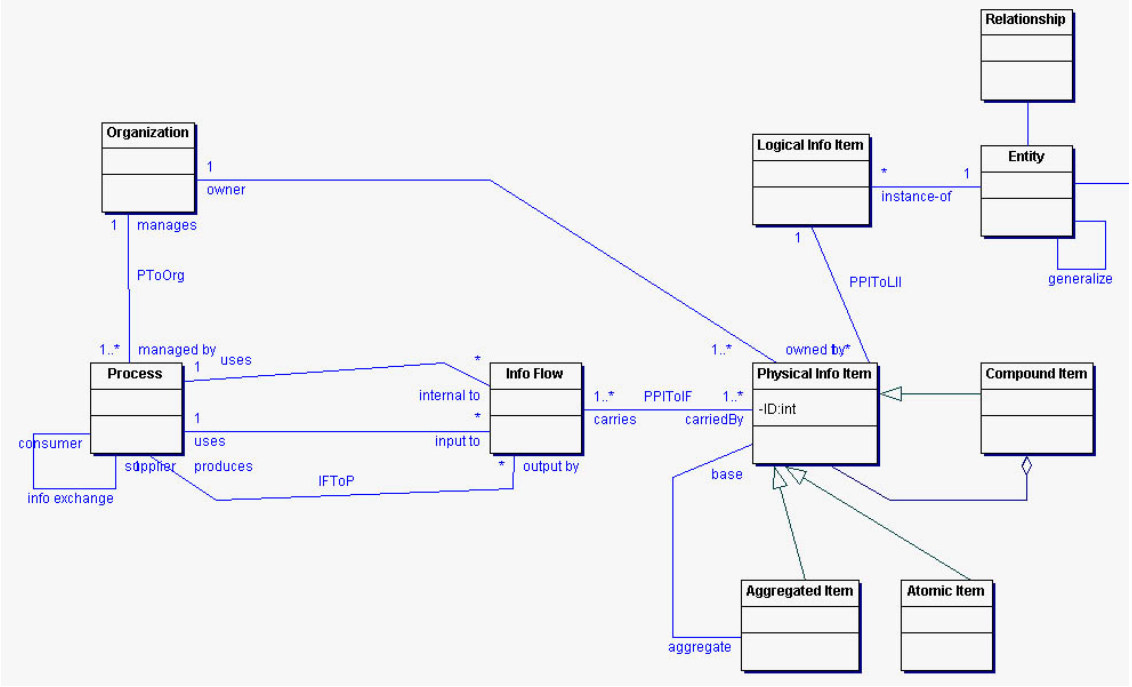

Figure 1. Information Items Data Model

An Information Flow *f* is a sequence of *physical information items* (PII), denoted *items(f)*, that are streamed from a supplier process to one or more consumer processes. For instance, given a domain entity called "Address", and its instance "J.Smith's address" (suitably identified using keys defined for Address), a PII would be a specific copy of J.Smith's address, that is produced at a particular time by a process p_1 and sent to a process p_2 over flow *f*. Thus, as many PIIs as are actually produced by any process at any time, are associated to the single *Logical Information Item* (LII) "J.Smith's Address". We denote the LII corresponding to a PII *i* as *lii(i)*. We assume that each PII is assigned a timestamp identifier that is unique within the scope of an Information Flow, making an explicit representation of time unnecessary.

In this work, we associate quality features to physical items rather than to logical items, because it is at the physical level that data errors occur. Consider for instance two copies of J.Smith's Address (two different address would also do). The first contains a spelling error at the time it is pulled from a database and sent to a different process. Then the error is corrected, and the new version is subsequently sent again. Because the quality level perceived by the receivers in the two instances differ, we are compelled to make a distinction between the two. Associating quality features to each PII output by a process accomplishes this.
We also assume that the organizations interested in exchanging instances of shared domain entities, have agreed on a common, integrated schema that defines those entities. The diagram in the figure shows part of the Entity-Attributes-Relationship metamodel for the domain entities.

The distinction that we have made between logical and physical items also holds for compound and aggregated items. As the diagram shows, a compound item is obtained recursively from other compound or elementary items (eg an Address may be composed of Street, City, ZIP code, etc.). An aggregated item is obtained from a collection of base items by applying some aggregation function to them (eg the average income of tax payers in a given town). Although aggregation and composition can be applied at the logical level, quality considerations once again lead us to consider composition and aggregation only at

the physical level. Consider for instance a process that takes a customer Name and Address from two different input flows, combines them with an internal flow consisting of customer orders, and produces an output flow of Invoices that can be mailed to customers. Although Name, Address and the compound Invoice item are defined in the logical model, the quality features of a physical Invoice can only be computed from the quality features of physical Name and Address. Similarly, the quality of an aggregate can be derived from the quality of its base items considered at the physical level. This corresponds for instance to measuring the accuracy of the average income of taxpayers, as opposed to that of individual taxpayers.

Quality dimensions and Descriptors

The quality features associated to PIIs are traditionally described in terms of *quality dimensions*. We refer the reader to the existing literature on the definition and use of quality dimensions, such as [28][23].

Given a set of quality dimensions *QDimSet*, a **Quality Descriptor** $QD_d(i)$ for dimension $d \in QDimSet$ and Physical Information Item i is a value $r \in D$ called the *quality rating* for i, ranging over values in the dimension-specific domain $D = D_0 \cup \{\text{UNKNOWN}\}$. The UNKNOWN value accounts for unavailable ratings.
The domain D may be discrete or continuous. When correctness is defined over the boolean domain $D = \{0,1\} \cup \{\text{UNKNOWN}\}$, the descriptor for correctness is written simply as: $QD_{corr}(i) = (r,c)$ where
r = 0 if i is incorrect, 1 if i is correct, and UNKNOWN if the rating is not available, and c is the confidence associated to the rating. For data obsolescence, the rating domain is the set of positive reals, and $QD_{obs}(i) = (r,c)$ where r is defined as above.

A rating value may be generated using various quality measurement techniques. Because in many practical circumstances computing reliable values for quality ratings on some dimensions may be difficult, the confidence value is used to express the uncertainty that is normally associated to the measurement. For instance, when a data item belongs to a homogeneous dataset, it is common practice to compute estimates of its quality rating on a sample extracted from the dataset. In this case, the confidence value accounts for the uncertainty in the estimate.

Each PII produced by a process has a Quality Descriptor associated to it. When the PIIs are obtained through aggregation and/or composition from otherPIIs, the QDs themselves are computed as functions of elementary QDs. For instance, a PII consisting of a batch of compound address records may have a single QD associated to it, that has been computed as part of the production process undergone by the PII itself, possibly starting from the descriptors for the base items. It is important to point out that our framework is designed to aggregate over sets of quality descriptors *after* they have been created by production processes. For the purpose of quality analysis, we are not concerned with the functions used internally by a process to compute complex QDs. However, we also note that aggregation functions are defined similarly both on the data domain (i.e., to compute descriptors for aggregated information items), and on the quality domain (i.e., to compute descriptors from an aggregation of base descriptors). Therefore, we deal with both problems in a uniform way in a later Section.

THE QUALITY CUBE

Building on the definitions just given, in this section we formally introduce Quality Cubes and derive a definition of Quality Offer and Demand.

Quality Profile Model

We model the quality profile of an organization as a *data cube* on a given set of dimensions. The intuition is to view a single item's quality profile as one point in a multidimensional cube, whose axes include a hierarchy of entities consisting of Information Items, flows, processes, organizations, and quality dimensions. The information carried by each Quality Point in the resulting *quality cube* is the single quality measurement at the finest level of granularity, i.e. the quality descriptor associated to a single physical data item and for a single dimension. Figure 2 shows the snowflake schema that has Quality Points as its "fact" entity.

The quality profiles for information flows, processes and for an entire organization can be computed as appropriate views from a base quality cube. Thus, once an appropriate set of aggregation functions is defined over quality descriptors, quality profiles at each level of granularity within an organization can be described within an established framework for multidimensional data.

In the following, we model quality cubes using the multidimensional database model proposed in [1], that we are going to summarize briefly below. The model includes a simple definition of data cube and of a small set of operators over the cube. The operators trasform cubes into other cubes, and thus they can be composed to produce complex cube transformations.

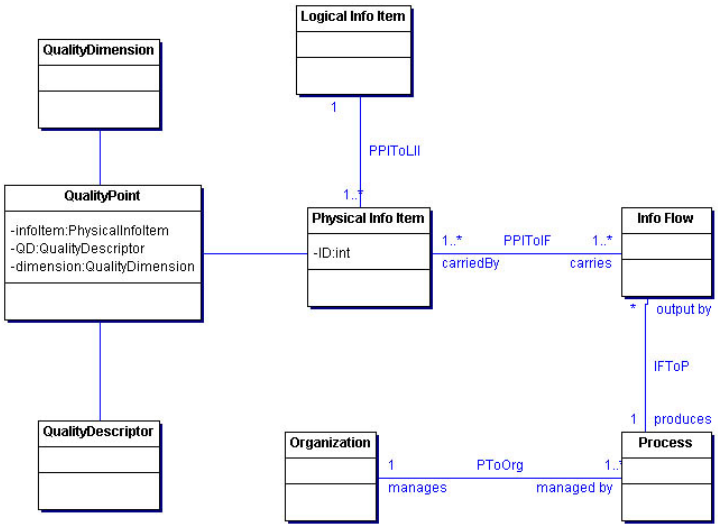

Figure 2. Quality Profile Model

In order to describe our schema using the formalism of the model, we start by naming the set of one-to-many relationships among hierarchical dimensions shown in the figure, as follows:

- **from PII to LII:** (*phys, logical*) ∈ PIIToLII iff *lii(phys) = logical*;
- **from PII to InfoFlow:** (*phys, f*) ∈ PPIToIF iff *phys* ∈ *items(f)*;
- **from Flow to Process:** (*f, p*) ∈ IFToP iff *f* ∈ *out(p)*;[2]
- **from Process to Organization:** (*p,o*) ∈ PToOrg iff *p* ∈ *proc(o)*.

We now recall the definition of cube given in [1]. A cube *C* is a pair $(\{D_1,...,D_n\},E_C)$ consisting of a set of *n* named dimensions D_i, with values in the set dom_{Di}, and a mapping E_C from a n-uple of dimensions into a set of elements. An element can be a boolean value, or a tuple $[x_1,...,x_n]$ of dimension values. To illustrate, let us define our base quality cube as:

$C_0 = (\{qd, pii, QD_{dim}(pii)\}, E_{C0})$, where

[2] Note that only output flows from a process are considered.

$E_{C0}([qd,x,QD_{dim}(x)]) = 1$ iff descriptor qdescr = $QD_{dim}(x)$ is defined}

where *dim* ∈ *QDimSet*. This model is *symmetric* in that *qdescr*, our measure of interest, is no different from any other dimension. In fact, unlike other asymmetric models in which the measures are defined in the schema, in this model aggregation functions can be potentially defined on any of the dimensions. To make the cube resemble our *asymmetric* schema, we apply the **push** operator to C_0, to obtain the new cube QP_0 whose elements consist of a 1-uple of the form [*qdescr*]:

QP_0 = push(*C0, qdescr*), where

$E_{QP0}([dim,x]) = qdescr$ if *qdescr* = $QD_{dim}(x)$ is defined, and 0 otherwise.

The elements mapping E_C now corresponds to the Quality Point "facts" entity.
Without loss of generality, we will assume that $QD_{dim}(x)$ is indeed defined for each PII *x* and each dimension *dim*, and use UNKNOWN values to fill in for the missing descriptors. In practice, the actual ratio of non-unknown QDs to the total number of items produced is expected to be small, reflecting an organization's quality measurement policy. While in an actual implementation the overwhelming size of the cube and the sparsity of its significant QDs are real issues, we choose to postpone such operational concerns until a later stage in our work.

From the base quality cube QP_0, a variety of interesting cubes can be derived by *aggregating* over the descriptors and along different axes and by *rolling up* along the dimensions hierarchies, yielding indicators for the quality of particular views over the overall population of items produced by a set of organizations. In the cube model we have chosen, the two additional operators we need in order to define those derived cubes are **restrict**, used to define a cube as a "slice" of another cube, and **merge**, that combines a roll-up operation along multiple hierarchies, with aggregation over the cube elements. Their general form is the following:

- **restrict(***C, D, P***)** removes from *C* the elements corresponding to dimension *D* that do not satisfy the condition expressed in predicate *P*;
- **merge**(*C*, f_{aggr}, {[D_1,f_{merge1}], ..., [D_k,f_{mergek}]}) operates on *k* dimensions. For each D_i, it merges the values from dom_i into values defined at the next higher level in D_i's hierarchy, according to merging function f_{mergei}. Aggregation function f_{aggr} is then applied to each set of elements resulting from the merge over each dimension.

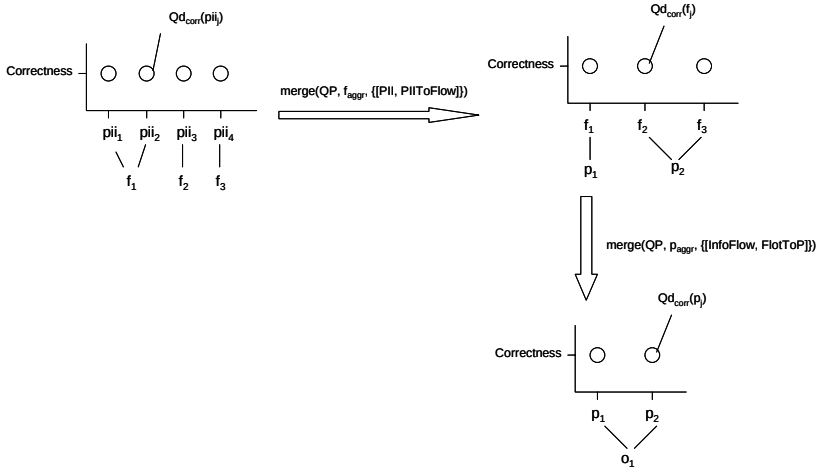

Figure 3. Example of use of the merge operator

An example of the use of the **merge** operator on a quality cube is shown in Figure 3, where a set of *ppi*s are rolled up along the common correctness value, to yield a new set of aggregated QDs, now defined at the flow level (pii_1 and pii_2 are aggregated beacuse they belong to the same flow f_1. In the next step, we apply merge again to roll-up from the flows to the processes, resulting in two new QDs that represent the aggregated correctness descriptors for processes p_1 and p_2.

The merge functions of interest in our model are expressed using the one-to-many relationships defined above between adjacent hierarchical dimensions in the schema, i.e., we define PPIToLII(*pii*) = *lii* iff (*pii*, *lii*) ∈ PPIToLII, and so forth.

Aggregation functions

Two types of aggregation functions are of interest. First, those functions that are associated to the aggregation and composition of physical items: they compute quality descriptors for aggregated and for compound physical items, given the quality descriptors of base items. Second, those functions that are associated to the roll-up operations in the quality cube: they compute quality descriptors for elements at a given level in the hierarchy of the cube dimensions, given the descriptors for the elements at the next lower level.

- Functions of the first type have the following general form:

  $QD_D(x) = f_{aggr}(\{QD_D(x_1),...,QD_D(x_n)\})$

  where $x_1,...,x_n$, x are physical items, and D is a single quality dimension. When x is an aggregation (i.e., a "set-of" Address items collected in a file that is sent over a flow as a single item), the x_i are homogeneous, i.e., $lii(x_i) = lii(x_j)$ for all i,j. Conversely, when x is a compound item, for instance an Address formed from City, Street, and ZIP code, the x_i may be heterogeneous. These functions are used by processes to associate a single descriptor to the individual physical items that are produced.

- Functions of the second type are used to manipulate quality cubes. Their general form is:

  $QD'_D = f_{aggr}(\{QD_{D1},...,QD_{Dn}\})$

  where $QD_{Di} \in E_C([D, x])$. From this general form, we define the following specific functions depending on the type of data point x: f_{aggr_PII} for x ∈ PII, f_{aggr_f} for x ∈ Info Flow, and f_{aggr_p}, f_{aggr_o}, for processes and organizations, respectively.

Only a generic description of these functions can be provided in the framework, while their specific definitions is best left to each framework instance, because they depend on the choice of an actual set of quality descriptors. In fact, we claim that the choice of suitable aggregations is critical to the success of the quality cube approach in a specific cooperative environment. For the sake of illustration, the following enumeration offers examples of aggregation functions of the second type that are not domain-specific. We distinguish among the following the *type of domain* on which the dimension *D* is defined:

- ***dom*$_D$ boolean:** The usual n-way boolean operators OR, AND, XOR, etc. yield appropriate (boolean) aggregated values that are useful eg to express correcteness. When correctness values are expressed as [0,1], other common (non-boolean) aggregations include the *density*, sum()/count(), of correct items, the correct/incorrect ratio, and so forth;
- ***dom*$_D$ discrete:** discrete domains are commonly used for subjective quality dimensions whose values should offer an immediate intuition to a human information consumer. For instance, the values of "usefulness" can be set to {low, mid, high} for intuitive understanding. In this case, a useful aggregation would yield a summary *histogram* of the distribution of values by their frequency over the population. This descriptor is obviously not closed wrt any discrete domain;
- ***dom*$_D$ continuous:** Any standard aggregation function on reals such as max(), min(), avg() and so forth are applicable. An example continuous dimension is obsolescence.

Quality Offer and Demand and Quality Adequacy

Using the basic formal machinery introduced above, we can now easily define the following fundamental quality profiles:

1. Quality profile at the flow level:

   QP_F = merge(QP_0, f_{aggr_PII}}, { [PII, PPIToIF] })

The profile for a specific flow is: $QP_F(f)$ = restrict(QP_F, InfoFlow = *f*). For instance, if *f* contains physical Address items, $QP_F(f)$ is the cube whose elements are the descriptors for all the Address items produced by all processes that generate *f*.

2. Quality profile at the LII level:

   QP_{LII} = merge(QP_0, f_{aggr_PII}, { [PII, PPIToLII] })

   The profile for a specific logical item *i* is: $QP_{LII}(i)$ = restrict(QP_{LII}, lii = *i*). For instance, if *i* is an Address item, the $QP_{LII}(i)$ cube contains the descriptors for all physical Address items.

3. Quality profile at the Process level:

   QP_P = merge(QP_F, f_{aggr_F}, { [Info Flow, IFToP] })

   and for a specific process *p*: $QP_P(p)$ = restrict(QP_P, Process = *p*). Again, this is the cube whose elements are descriptors for all physical items produced by process *p* (over any information flow that originates from *p*).

4. Finally, the quality profile at the Organization level is defined as:

   QP_O = merge(QP_P, f_{aggr_P}, { [Process, PToOrg] })

   As Figure 3 shows, this cube contains one descriptor for each Organization *o* and for each Quality Dimension *d*, representing a summary of the quality offered by *o* for dimension *d*.

We now use these fundamental profiles to introduce the notions of **Quality Offer** and **Quality Demand**, that characterize the relationships between information supplier organizations (the producers), and information customers organizations (the receivers):

Quality Offer. The *quality offer* associated to a supplier *s* is the base cube QP_0 restricted to *s*. The cube contains the entire "quality history" for a supplier with respect to the information items it produces. As we have seen, from this base cube more concise cubes that are useful in practice can be derived. The intuition is that, using the quality offer cubes provided by suppliers along with the information items themselves, customers may make informed decisions regarding which item is best acquired from which organization, and through which process;

Quality Demand. Complementary to offer, quality demand consists of a set of **quality conditions** expressed by an organization on the elements of a quality offer cube, i.e., predicates defined on quality descriptors, that must be true in order for the information provided by the supplier to be of acceptable quality to the customer.
Suppose for instance that an organization *o'* defines a requirement for Address items provided by organization *o*, stating that at most *X*% of incoming addresses supplied by a process $p \in proc(o)$ are allowed to be incorrect. This is a reasonable requirement in the common case when the customer is bound to obtain its information from *o*, but may choose which process it is going to get the information from. In this case, we say that an Address item is **fit for use** by *o'* if it can be proven that its correctness quality descriptor satisfies the condition stated in the requirement. Now, suppose that restrict($QP_P(p_1)$, PII = Address}) and restrict($QP_P(p_2)$, PII = Address) for processes $p_1, p_2 \in proc(o)$ both include values for the correctness dimension. Then:

- for each of the two processes, the requirement condition can indeed be evaluated, and thus the fitness for use for the Address by *o'* can be established in each of the two cases;
- if the aggregated correctness values (eg average correctness) can be ordered, then the processes themselves can be ordered based on the measured distance between the threshold value set in the requirement, and the value that appears in the descriptors. Thus, we may say that one of the two processes is more or less **adequate** than the other from the point of view of *o'*.

Quality demand requirements can be expressed more formally by testing the correctness value in the descriptor obtained through a suitable aggregation over the offer cube. To this end, we introduce a family of functions, denoted $conditioned_X()$, that compute new cubes containing the elements that are subject to

conditions, starting from the offer cube. The generic pedix *X* indicates different members in the $conditioned_X$ family of functions. Parameters are left implicit to simplify the notation. In the following example, $conditioned_{x0}(QP_0, o)$ computes a cube that contains the quality offer for a specific LII (Address) and dimension (Correctness) for different processes in *o*:

Let $QP^{(1)}$ = merge($QP_O(o)$, f_{aggr_PII},{ [PII, PPIToLII]}), and $QP^{(2)}$ = restrict($QP^{(1)}$, LII = *Address*).

Then:

$conditioned_{x0}(QP_0, o)$ = restrict($QP^{(2)}$, qDim = Correctness).

Each element in the resulting cube now contains an aggregated quality descriptor whose value depends on the definition of f_{aggr}. Hence, the type of condition that can be expressed depends on those functions as well, since it must be consistent with the value contained in the descriptors. In the example, the condition "at most *X*% of incorrect records" assumes that the cube elements now contain the average number of incorrect addresses for each process, aggregated over all PIIs and over each flow. More precisely, we define quality requirements QR for customers as a set of pairs of the form:

$QR = \{[conditioned_{x1}, C_1], ..., [conditioned_{xk}, C_k]\}$

where functions $conditioned_{Xi}$ are defined using the multidimensional operators introduced so far, assuming the same definitions for all f_{aggr} and f_{merge} functions, defined at page 8. In other words, information customers and suppliers agree on a common instance of the framework we are describing. Each of the conditions C_i is a predicate defined on the elements of cube $conditioned_{Xi}(QP)$, for a particular quality offer cube *QP*.

As a further example, note that sometimes it is useful to test a condition on aggregated descriptors, under the additional constraint that those descriptors be representative of a sufficient number of underlying elementary descriptors. Using the formalism just introduced, one can easily test the *density* of the significant (i.e., non-unknown) quality descriptors within a cube *C*, using an expression like merge(C,$f_{density}$,{}), where aggregating function $f_{density}$ computes the ratio between the count() of elements with value ≠ UNKNOWN, and the overall elements count(). Because no merge function is defined, the count is computed over the entire set of elements in the cube.

Having presented this formalization of the notion of quality demand, we make our definition of fitness for use more precise. We say that a set of items are **fit for use** relative to a customer quality requirement *QR*, if, whenever its quality values appear in a cube defined by a $conditioned_X$ expression, the corresponding conditions are satisfied.

Along with Demand, we want to define the **adequacy** of a supplier, in such a way that multiple potential suppliers may be ranked according to their different degree of adequacy with respect to a Demand. In order to do this, to each *QP* we associate a value *score(QP)*, a function of the values in the quality profile. The specific definition of score depends on the customer's business rules. For instance, if v_1, v_2 are the numeric values of the QDs for correctness and currency for a given organization at the process level in a profile *QP*, then the score may be computed as a linear combination $\alpha v_1 + \beta v_2$. Or, for discrete quality values, one definition could be *score(QP)* = 1 if correctness = high, 2 if (correcteness = low and currency = high). The score is a synthetic measure of quality values that can be used to rank suppliers with respect to quality demand.

To introduce adequacy, let us assume that the conditions *C* appearing in each quality requirement *QR* are of the special form $v_i \geq c_i$, where the c_i are constants. The *minscore* associated to *QR* represents the minimal acceptable quality level, and is defined as the score function computed using the set $\{c_i\}$ in place of the values $\{v_i\}$ found in the *QP* cube. In the examples above, the minscore would be $\alpha\ c_1 + \beta\ c_2$. In this case, the {adequacy} of a supplier with associated profile *QP* is defined as *score(QP) - minscore*.

This simple definition of adequacy may be generalized by considering other types of conditions and scoring functions. Using adequacy, a customer may rank its potential information suppliers.

ESTABLISHING QUALITY ADEQUACY IN CIS

We are now going to build upon the notions of Quality Profile and of Quality Offer and Demand, in order to sketch the essential elements of a methodology for Information Quality improvement in a cooperative environment.

A *Quality State* is a summary of the quality profiles exhibited by each CIS in a federation, along with the description of all the processes managed by each organization. With respect to the data and quality models introduced in the previous section, a Quality State is defined by an overall Quality Cube, plus an instance of the data model introduced at the beginning of the paper, i.e., a description of organizations, their processes, the inter-process relationships expressed by information flows, and the logical data items carried by the flows. We call the latter the current *CIS structure*.

We model the process of quality improvement as a sequence of transitions in the space of all quality states, given a starting state and a goal state. Transitions represent the application of one or more quality-improving operators, that transform both the CIS structure and its associated Quality Cube. We may state the general problem of quality improvement in CIS as the problem of finding a suitable sequence of transitions from the start state to the goal state, subject to a set of constraints that may limit the applicability of the transition operators.
In this section, we outline the basic elements of the model, and provide an initial insight into a methodology for goal-oriented state space traversal.

A model for describing the target state of a CIS federation

We identify four major classes of processes, that are useful to define process-transformation operations, as follows:

- **PInt**: processes that are internal to each organization. An example is a process that updates a database controlled by the same organization;
- **P2P**: processes that establish a relationship with other organizations. These processes are typically responsible for the exchange of information flows across organizations;
- **PExt**: processes that establish a relationship with external users. These processes carry out the I/O necessary to communicate with the users of an organization's information systems;
- **QP**: *quality processes* that contribute to the IQ management within an organization. Typically, these processes are only introduced into the organization to support a quality management plan, while they are not functional to the organization's core business processes.

We can now make the definition of quality state given earlier more precise. For a organization *o*, we identify three quality elements, as follows:

- the *Current Offered Quality Profile*. This is the same as $QP_O(o)$ according to our formal definition;
- the *Current Input Quality Profile*. This is the Quality cube consisting of all the quality descriptors for each PII in each *incoming* flow into each process $p \in proc(O)$. Note that, while $QP_O(O)$ refers to the quality *produced* by *O*, this is the description of the current quality *obtained* by *O* from other organizations. We may obtain a formal definition for this cube simply by redefining the IFToP relationship given earlier (see "Quality Profile") as: $(f, p) \in$ IFToP iff $f \in in(p)$. This corresponds to replacing the "produces/output by" relationship between Process and Info Flow in Figure 3, with a "receives/input by" relationship;
- the *Target Quality Demand* (defined at the end of the previous section) for each process $p \in proc(o)$.

In addition, a quality state includes the current *CIS structure*, already defined.
A *target state* is any state in which, for each organization *o*, its Target Quality Demand is satisfied by the

Current Input Quality Profile. This means that either there exists an organization *o'* such that is *adequate* with respect to *o*, or there exists a set of organizations, each providing some of the flows required by *o*, such that the union of all flows is adequate to *o*.

Note that, in the state space, the Target Quality Demand for *o* is constant, while the Current Input Quality Profile for *o* changes depending on the selection of different information providers, and the Current Offered Quality Profile usually changes as a consequence of transformation internal to the organization or to its input flows.

Thus, state transitions are caused by the application of elementary operations that alter the CIS structure of one or more organizations, and hence they modify one or more quality profiles. Our goal is to select operations that lead to a target state. We identify the following types of elementary operators:

- *process-based operators* that add, alter or remove processes. These operations correspond to performing process re-engineering at both the organization and the inter-organization levels;
- *data-based operators* that add, alter (modify the LIIs) or remove a flow between processes within the same organization, or a flow between new or existing P2Ps in different organizations. These operators may also export previously private data item (i.e., defining a new LII from a private data item, and attaching it to a flow), and viceversa, they may remove LIIs from flows, making them private. Flows are usually removed when they carry low-quality data, and they are added to enable new or existing processes to carry out additional functionality.

Note that the simple operators just introduced can be used to implement decisions taken as part of a re-engineering effort. For instance, they can be used to ``rewire" the network of processes so as to allow an organization to select one of potentially multiple flows that carry the same or similar information, each offering a different quality profile.

Quality Services and quality-based operators

In addition to the simple operators just described, we also introduce operators that enable organizations to access and use *Quality Services*. These services may be offered as part of the CIS environment and affect in a number of ways the IQ manager's ability to implement quality improvement strategies. In order to use these services, organizations usually must introduce new processes, that we call *quality processes*. We describe four quality services, along with examples of quality processes required to use them.

Quality Certification. This service creates *quality metadata* (eg of the form used to define Quality Descriptors, QD) and attaches a quality certificate to PIIs carried over a flow. The certificate contains the QD, along with other non-quality metadata such as creation and latest update timestamps, data originator, and possibly more.

This service may be implemented using a peer-to-peer model using quality processes that are local to each data-originating organization. Working in coordination with existing PExt, they attach the metadata to outgoing PIIs within a flow, and conversely, they decode and interpret the metadata upon arrival of incoming PIIs. Additionally, the decoded metadata may be used to enforce local quality policies concerning the use of the incoming data (eg low-quality data may be rejected);

An event-driven *notification service* based on the publisher/subscriber model can be used as an additional information flow channel to exchange both data and its associated quality metadata. Using this service, subscriber organizations receive notifications about variations in data values and quality metadata that are published by other organizations. A subscriber organization may use this service to monitor changes in other copies of its data that exist elsewhere in the federation, and symmetrically, it may post updates that just occurred on its own copy.

This service is usually implemented using a central hub that receives events from publishers and forwards them to the event's subscribers. Thus, the quality processes associated to this service consist simply of adapters that enable the send/receive operations. Additionally, however, these quality processes are responsible for the handling of incoming events, based on local business rules. A typical rule would call for an update in a local database whenever new values for a data item are received through an event;

- A *brokering service* for information quality data and metadata may hold a central registry of data sources, and coordinates access to specific data items to interested parties. With reference to the last scenario above, an organization may use this service to locate reliable sources to reference information, and to obtain copies of data items that conform to specific quality requirements (i.e. of currency, format consistency, accuracy, etc.);
- A *Data Source Trust Service* can be offered for *rating* the quality of information sources. The idea, already familiar in some e-commerce and online auction applications, is to associate reliability metrics to information producers, rather than, or in addition to, the information itself. Using a rating system, parties that are interested in obtaining items from other CIS network members may consult the service to determine the expected quality of incoming items. For example, one rating model may use a producer's past performance, to determine its current level of trust. Depending on the model, past performance levels may be assessed by CIS peers through a voluntary voting system, or it may be determined by a rating authority;
- A *Quality Validation Service*, whose purpose is to provide clients with an online quality assessment, expressed using Quality Descriptors, for a given data item. For instance, a City organization that is in charge of residents' data may provide quality assessment on currency and accuracy data about citizens (to authorized parties).

Rules of cooperation and a black box description of the methodology

The quality improvement methodology consists of guidelines that help Information Quality Managers operate on the state space, in order to reach a target state from a given initial state. The main output produced through the methodology is a specific quality improvement plan, i.e., the coordinated application of quality improvement operators.

The operators we have introduced assume that local IQ managers have access to each organization's own information assets (i.e., their conceptual schema). This information is used to determine which private LIIs may be potentially exported through new flows.
In addition, we introduce an important set of constraints, called *cooperation rules*, that limit the way the transformation operators are used. In a cooperative scenario, achieving the common goal of improving information quality federation-wide requires organizational changes both at the business and at the system levels, through the application of quality improvement operators. However, these changes may result in a violation of an organization's autonomy. For instance, a plan that calls for the business process re-engineering of some internal processes (of type PInt), may be considered intrusive by one of the organizations that are affected.

Hence, cooperation rules are introduced so that each individual organization may preserve in part its autonomy when it participates to a common IQ improvement effort. These rules limit the use of theoperators to those that each organization finds acceptable.
We state the form of rules using an informal grammar, as follows:

<op> [is | is not] allowed for [<element> | all elements] [if <conditions>]

where <op> is one of the transformation operators, <element> is an instance of one entity of the federation data model (i.e. organization, process, flow, LII) and <conditions> are predicates on the values of instances of the data model. More complex constraints can be expressed using logical connectors.
In this paper, we are leaving the scope of the rules deliberately vague. For instance, we would like to be

able to express constraints like the following:

- "the values of a given LII may not be exported to organizations o_1, o_2". This constraint may be formalized using our framework (omitted), considering the allowed/disallowed flows that may carry the PIIs corresponding to a given LII;
- "a specific set of internal processes in org *o* may not be altered or removed";
- "LIIs *a* and *b* must be obtained from the same source". This constraint predicates on the processes that originate the flows that carry the given LII;
- "org *o* is not going to attach quality metadata to its outgoing PIIs". Quality metadata is attached to data by suitable quality processes. This constraint states that *o* is not interested in using such quality processes (for all elements, unconditionally).

A summary of the inputs to the methodology is shown in Figure 4.

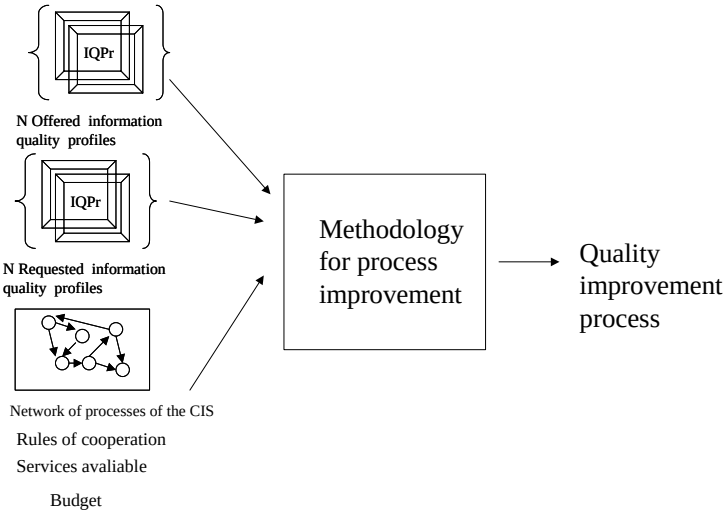

Figure 4. Schematic of an inter-organization process improvement methodology

We assume that the quality improvement process concerns a number *N* of federated CIS organizations, that are characterized by *N* offered/requested quality profiles, an overall network of communicating processes, cooperation rules and available quality services.
From this general scenario, we may derive several simpler scenario, eg in which only one CIS organization requires quality improvement, or else in which only a subset of all the flows are considered.

FURTHER WORK

In this paper, we have defined the notions of quality profile associated to a CIS, and we have presented a formal framework to define quality demand and quality offer in the context of cooperative processes. We are currently working on a prototype to demonstrate the feasibility of the approach, and on its application to a real case study.
Our approach is really about monitoring and warehousing quality data as application data flows across information systems, and then performing OLAP on it with the purpose of matching offer and demand. This approach relies on the key assumption that quality data can in fact be measured and obtained in a systematic way. Our first challenge is to propose a system architecture that makes this essential task possible and inexpensive.

Then, we need to gain further insight into quality-oriented services for CIS and the way they help supplier organizations match quality demand. Ultimately, we hope to show how our framework can help solve the typical problem of an Information Quality Manager, i.e., finding within a given budget an optimal process

of quality improvement.

REFERENCES

[1] R.Agrawal, A.Gupta, and S. Sarawagi, Modeling multidimensional databases, In A. Gray and P. Larson, eds, *Procs. ICDE*, April 7-11, 1997, Birmingham U.K, pages 232--243. IEEE Computer Society, 1997.

[2] Y. Akao, ed. *Quality Function Deployment: Integrating Customer Requirements into Product Design*. Productivity Press Inc., 1990. ISBN: 0915299410.

[3] D.Ballou, R.Wang, H.Pazer, and G.K.Tayi. Modeling information manufacturing systems to determine information product quality. *Journal of Management Sciences*, 44(4), April 1998.

[4] P.Bertolazzi, M.Scannapieco. Introducing data quality in a cooperative context. In *Procs. IQ 2001*, Boston, MA, 2001.

[5] F.Caruso, M.Cochinwala, P.Missier et al. Telcordia's database reconciliation and data quality analysis tool. In Amr El Abbadi et al, editors, *VLDB 2000*, Proceedings of 26th International Conference on Very Large Data Bases, September 10-14, 2000, Cairo, Egypt, pages 615--618. Morgan Kaufmann, 2000.

[6] DaQuinCis project, at http://www.dis.uniroma1.it/~dq/docs.html

[7] E.W.Deming, *Out of the crisis*. Center for Advanced Engineering Study, MIT, Cambridge, MA, 1986.

[8] G. De Michelis, E. Dubois, M. Jarke, F. Matthes, J. Mylopoulos, M. Papazoglou, K. Pohl, J. Schmidt, C. Woo, and E.Yu, Cooperative information systems: A manifesto. In Mike P. Papazoglou and Gunther Schlageter, editors, *Cooperative Information System: Trends and Directions*. Academic Press, 1997.

[9] L.P. English. *Improving Data Warehouse and Business Information Quality: Methods for Reducing Costs and Increasing Profits*. John Wiley \& Sons, 1 edition, March 1999. ISBN: 0471253839.

[10] I.P. Fellegi and A.B. Sunter. A theory for record linkage. *Journal of the American Statistical Association*, 64, 1969.

[11] M.G.Fugini et al, *Data quality in cooperative web information systems*. Personal Communication, 2002.

[12] H.Galhardas, D.Florescu, D.Shasha, and E.Simon. An Extensible Framework for Data Cleaning.

[13] In *Proceedings of the 16th International Conference on Data Engineering* (ICDE 2000), San Diego, CA, USA, 2000.

[14] J.M.Juran, F.M.J.Gryna, and R.S. Bingham. *Quality Control Handbook*. McGraw-Hill Book Co., New York, 3rd edition, 1974.

[15] M.A. Hernandez and S.J. Stolfo. Real-world Data is Dirty: Data Cleansing and The Merge/Purge Problem. *Journal of Data Mining and Knowledge Discovery*, 1(2), 1998.

[16] H.B.Kon R.Y.Wang and S.E. Madnick. Data Quality Requirements: Analysis and Modeling. In *Proceedings of the 9th International Conference on Data Engineering* (ICDE '93), Vienna, Austria, 1993.

[17] M.Mecella, M.Scannapieco, A.Virgillito, R.Baldoni, T.Catarci, and C.Batini. *Managing data quality in cooperative information systems*. In Proceedings of the 10th Internation Conference on Cooperative Information Systems, Irvine, CA, 2002.

[18] P.Missier, G.Lalk, V.S. Verykios, F.Grillo, T.Lorusso, and P.Angeletti. Improving data quality in practice: A case study in the italian public administration. *Distributed and Parallel Databases*, 13(2):135--160, 2003.

[19] J. Mylopoulos and M.P. Papazoglou, Guest editor's introduction: Cooperative information systems. *Intelligent Systems*, 12(5), September/October 1997.

[20] H. B. Newcombe, J. M. Kennedy, S. J. Axford, and A. P. James. Automatic linkage of vital records. *Science*, 130:954--959, October 1959.

[21] P.S. Pande et al, *The Six Sigma Way: How GE, Motorola, and Other Top Companies are Honing Their Performanc*e. McGraw-Hill Trade, April 2000. ISBN: 0071358064.

[22] Rath & Strong's six sigma pocket guide. Rath & Strong, Inc., October 2000. ISBN: 0970507909.

[23] Y.Wand and R.Wang. Anchoring data quality dimensions in ontological foundations. *Communications of the ACM*, volume 39. ACM, 1996.

[24] Y.R.Wang and S.E.Madnick. A polygen model for heterogeneous database systems: The source tagging perspective. In *16th VLDB Conference*, pages 519--538, Brisbane, Australia, 1990.

[25] R.Y.Wang, M.P.Reddy, and H.B.Kon. Toward quality data: An attribute-based approach. *Decision Support Systems*, 13:349--372, 1995.

[26] R.Wang et al. *IP-MAP standards and guidelines*. Internal draft, February 2002.

[27] R. Wang, A product perspective on total data quality management, *Communications of the ACM*, 41(2), February 1998.

[28] R.Y.Wang, M.Ziad, and Y.W.Lee. *Data Quality*. Advances in Database Systems. Kluwer Academic Publishers, 2001.

[29] V.S.Verykios, M.G.Elfeky, A.K.Elmagarmid, M.Cochinwala, and S.Dalal. On the accuracy and completeness of the record matching process. In Sloan School of Management, editor, *Procs. of Information Quality Conference*, MIT, Cambridge, MA, 2000.