A Quality Framework for Data Integration Incorporating User Requirements

Jianing Wang, Alexandra Poulovassilis, and Nigel Martin

Department of Computer Science and Information Systems,
Birkbeck College, University of London, London WC1E 7HX
{jianing,ap,nigel}@dcs.bbk.ac.uk

Abstract. The data integration (DI) process involves multiple users with roles such as administrators, integrators and end-users, each of whom may have requirements which have an impact on the overall quality of an integrated resource. Users' requirements may conflict with each other, and so a quality framework for the DI context has to be capable of representing the variety of such requirements and provide mechanisms to detect and resolve the possible inconsistencies between them. This paper presents a framework for the specification of DI quality criteria and associated users' quality requirements. This is underpinned by a Description Language formalisation with associated reasoning capabilities which enables a DI setting to be tested to identify those elements inconsistent with users' requirements.

Key words: Data Integration, Quality Assessment, Quality Metrics

1 Introduction

Historically and currently, data conforming to different formats are gathered and organised by different parties. However, different users may need to access such data sources according to their specific requirements. This may require redefining data into different formats, combining relevant data from different sources, and combining incomplete data sources in order to form a more complete view. Combining and transforming data from different data sources is a complex problem and is the aim of *Data Integration (DI)*. In the DI context, data conforming to different data models can be transformed and accessed through a *global schema* using *mappings* between this schema and the data sources. A typical DI setting can therefore be represented as a triple $\langle GS, LSs, M \rangle$, where GS is the global schema, LSs are the local (ie. data source) schemas and M are the mappings between GS and the LSs.
Assessing the quality of a DI setting is a complex task. The data integration process may involve multiple users with different roles, and their quality requirements may not be consistent, in the sense that the same integrated resource cannot satisfy all requirements and, therefore, either the integrated resource or the users' requirements need to be modified. A quality framework for the DI context has to be capable of representing

these varieties and provide mechanisms to detect and resolve the possible inconsistencies between the users' quality requirements.
With the need for these capabilities in mind, we have developed a quality framework supporting quality criteria such as completeness and consistency, metrics for measuring the extent to which an integrated resource satisfies the desired quality criteria, and the capability of representing the quality requirements of different users. Our quality framework is underpinned by a Description Logic formalisation, and users' quality requirements are specified as logic statements. This enables formal reasoning to be applied to validate different users' requirements, as specified from their different quality perspectives. It also allows reasoning to be applied to validate individual quality requirements with respect to the integrated resource.
This paper is organised as follows. Section 2 briefly reviews research relating to quality assessment in the DI context. In Section 3, we describe our quality framework for DI and show how this framework is formally represented in Description Logic. There follows an example relating to the Higher Education domain to demonstrate how the quality of an integrated resource can be improved using our approach. Conclusions and further work are discussed in Section 4.

2 Related Work

Previous research [1, 2] has indicated that users with different roles are important in the DI context. Such roles include administrators, integrators and end-users. Different users may define their own quality requirements from their different perspectives. Therefore such requirements need to be considered individually and also as a whole in assessing the quality of an integrated resource. Although existing DI tools are designed to assist in many integration tasks, DI is still a complex problem due to the heterogeneity of the data sources and the variety of the users' quality requirements. None of the current DI tools supports quality management functionality within the integration workflow, whereas in recent work we propose such a DI architecture and methodology [3].
Recently, some research has proposed methods for detecting the DI quality or the correctness of a DI setting directly or indirectly. The work in [4] defines several quality measurement methods in the DI context, relating to the schema completeness, schema consistency and schema minimality quality criteria. These methods are based on information extracted from the schema metadata level, such as the proportion of the concepts in the integration domain that are represented by the schemas. Other work [5] has presented an approach for mapping selection based on the comparison of instances between the source schema and target schema extracted via different mappings in a data exchange context. The approach in [6] determines the quality of collaborative tasks, such as data integration, with respect to the users' quality requirements through users' feedback; this is in contrast to users' quality requirements expressed as logic statements over a quality hierarchy in our approach. While not motivated

explicitly from a quality perspective, other techniques can be adopted for measuring the quality of integrated resources. Such work includes instance checking methods which may be used to validate and refine mappings such as [7, 8], constraint validation such as [9, 10], and mapping cores to generate the minimum set of mappings with respect to users' queries [11].

3 Our Approach

We propose a *Quality Framework for Data Integration (QFDI)* that is composed of four major parts, termed *ITEM*, *METRIC*, *QUALITY CRITERIA* and *USER*, as illustrated in Figure 1.

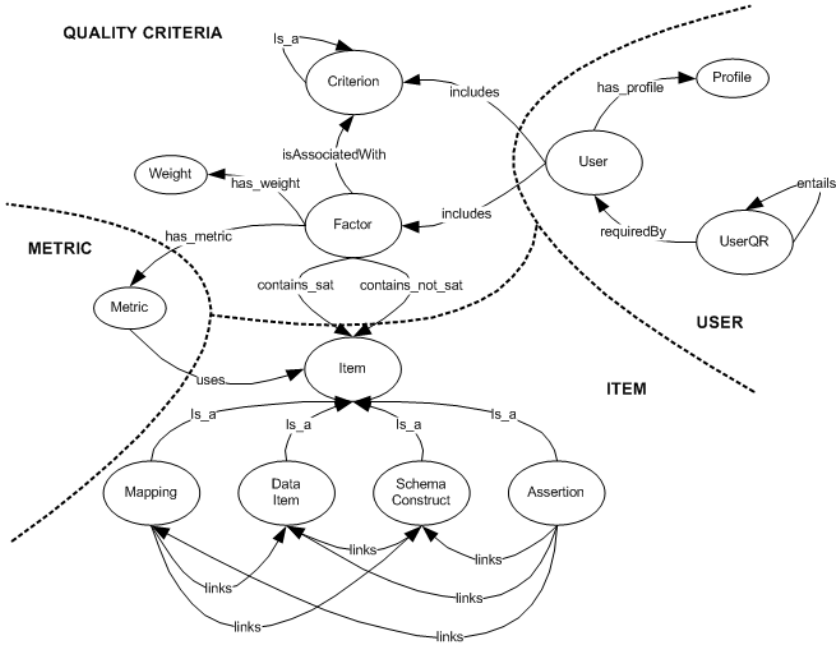

Fig. 1. The Quality Framework for Data Integration (QFDI)

ITEM contains the representations of knowledge extractable from the elements comprising a DI setting. By 'elements' we mean the fundamental constituents of an integrated resource, including *Data Item*, *Schema Construct*, *Mapping* and *Assertion*. Assertions are defined by integrators so as to express domain-specific knowledge relating to an integrated resource. All of these are represented as sub-concepts of the *Item* concept. Links exist between these sub-concepts, represented as the *link* property, to represent how the extent of one concept relates to that of another.
In the METRIC part, different measurement methods (metrics) are represented by the *Metric* concept. Each metric is defined over instances of

the Item concept in the ITEM part. The measurement results are stored as the extent of the Metric concept.
QUALITY CRITERIA contains the representation of the quality hierarchy as defined by the data integrator for a particular integrated resource. This hierarchy is built from two concepts, *Criterion* and *Factor*, and the relationships between them, namely *is_a* and *isAssociatedWith*. Each quality criterion can have many sub-criteria, linked by the *is_a* property. Each quality factor is associated with one quality criterion, using the *isAssociatedWith* property. Each quality (sub-)criterion can be associated with several quality factors. Each quality factor is associated with a quality metric in the METRIC part using the *has_metric* property. The *Weight* concept is associated with the Factor concept and indicates the specific weight of each quality factor as defined by the users.
The *contains_sat* and *contains_not_sat* properties link the Factor concept and the Item concept. These properties represent the DI elements that satisfy and that do not satisfy a quality factor, respectively. These two properties are disjoint.
USER contains the *User* concept. Users with different roles may have different quality requirements. For example, end-users may have requirements regarding the amount of information returned from a DI setting, whereas a data integrator may focus more on query performance requirements. Different user requirements are represented by the *UserQR* (User Quality Requirement) concept and they can be related by using the *entail* property, meaning that if the integrated resource satisfies one user requirement, the integrated resource also satisfies another user requirement that is entailed by the first one. The *Profile* concept represents the overall quality of the integrated resource with respect to a specific user, calculated as $w_1 \times r_1 + \ldots + w_n \times r_n$, where r_i and w_i are the measurement of a quality factor i and its user-specified weight, respectively.
So far we have investigated three quality criteria in the context of data integration: *completeness*, *consistency* and *accuracy*. Each criterion is categorized into several sub-criteria, and quality factors and their measurement methods are defined for each sub-criterion. We refer readers to [3, 12] for details of these quality criteria, factors and measurement methods. The quality criteria, factors and measurement methods that we describe there are not exhaustive. They are indicators of what is possible within our framework, and can be refined and extended in the future, following validation with real-world case studies and users.
Example. To illustrate our approach to DI quality assessment and improvement, we now introduce a simple case study (see Figure 2) composed of three data sources and a global schema. Database 1 (with schema LS1) contains detailed descriptions of the degree programmes and the staff of a university. Database 2 (LS2) contains detailed information about the undergraduate and postgraduate programmes taken by students who are enrolled on undergraduate courses. Database 3 (LS3) contains detailed information of the postgraduate programmes. A version of the global schema (GS) is created representing the head teachers of all programmes.
As described previously, users can state various quality requirements on an integrated resource based on their different perspectives. In our frame-

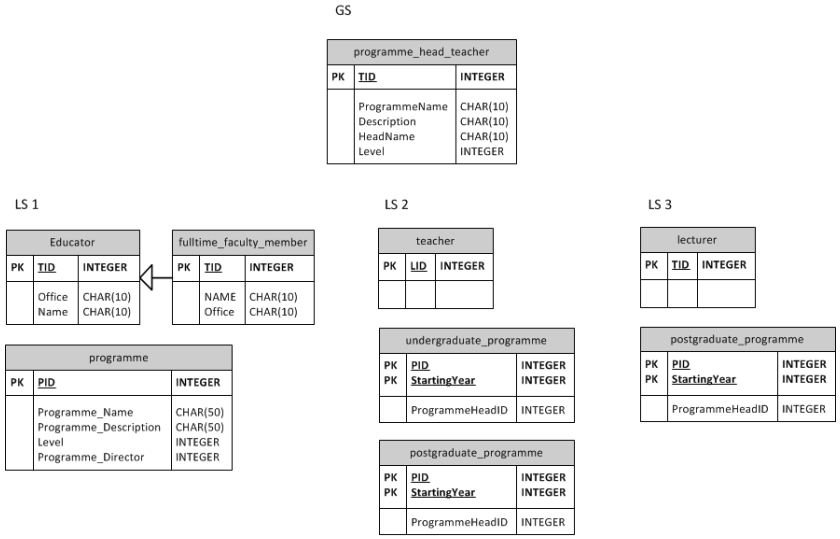

Fig. 2. Example Schemas

work, users' quality requirements are expressed as logic statements over the required quality factors. In providing the reasoning capabilities required by the QFDI, we use a Description Logic language. Description Logic (DL) is a family of formal knowledge representation languages based on the notions of *concepts* and *roles*. DL is characterised by constructors that allow complex concepts and roles to be built from atomic ones [13]. In particular, we adopt the $\mathcal{SI}$ DL, which provides sufficient expressive power for our reasoning purposes in the QFDI.

Terminology axioms are used to express the users' quality requirements. The terminology axioms we consider are *inclusion*, $C \sqsubseteq D$, and *equality*, $C \equiv D$, where C and D are concepts expressed in the syntax of the $\mathcal{SI}$ DL. More complex assertions can be created from these two basic ones plus negation.

Our quality framework is represented in the $\mathcal{SI}$ DL, where each oval in Figure 1 (except the USER part of the diagram) is represented as a DL concept named by the text in the oval and each link is represented as a role named by the name of the link. The DI elements are represented as individuals of the *Data Item*, *SchemaConstruct*, *Mapping* and *Assertion* concepts and are associated with quality factors via the *contains_sat* and *contains_not_sat* roles. Users' quality requirements are expressed as logic statements using terminology assertions, as explained above.

To illustrate, consider the simple case study introduced earlier and the following mapping that defines the TID and $ProgrammeName$ attributes of the table in the global schema GS as the union of three conjunctive queries over the source schemas LS1, LS2, LS3:

$\exists i, j, k.GS : programme\_head\_teacher(t, n, i, j, k) \leftarrow$
$\exists o, p, y.\ LS_1 : Educator(t, o, n) \wedge LS_2 : undergraduate\_programme(p, y, t)$

$\exists i, j, k.GS : programme\_head\_teacher(t, n, i, j, k) \leftarrow$
$\exists o, p, y.\ LS_1 : Educator(t, o, n) \wedge LS_2 : postgraduate\_programme(p, y, t) \wedge y > 1999$
$\exists i, j, k.GS : programme\_head\_teacher(t, n, i, j, k) \leftarrow$
$\exists o, p, y.\ LS_1 : Educator(t, o, n) \wedge LS_3 : postgraduate\_programme(p, y, t)$

	Satisfying Elements	**not-Satisfying Elements**
f_1	$\{LS_1\_Educator$, $LS_1\_tid_{Educator}, LS_1\_name_{Educator}$, $LS_1\_office_{Educator}$, $LS_2\_undergraduate\_programme$, $LS_2\_postgraduate\_programme$, $LS_2\_pid_{undergraduate\_Programme}$, $LS_2\_pid_{postgraduate\_Programme}$, $LS_2\_year_{undergraduate\_Programme}$, $LS_2\_year_{postgraduate\_Programme}$, $LS_2\_phid_{undergraduate\_programme}$, $LS_2\_phid_{postgraduate\_programme}$, $LS_3\_postgraduate\_programme$, $LS_3\_pid_{postgraduate\_Programme}$, $LS_3\_year_{postgraduate\_Programme}$, $LS_3\_phid_{postgraduate\_programme}\}$	$\{LS_1\_fulltime\_faculty\_memeber$, $LS_1\_tid_{fulltime\_faculty\_memeber}$, $LS_1\_name_{fulltime\_faculty\_memeber}$, $LS_1\_office_{fulltime\_faculty\_memeber}$, $LS_1\_Programme$, $LS_1\_pid_{programme}$, $LS_1\_progrmamme\_name_{programme}$, $LS_1\_programme\_description_{programme}$, $LS_1\_level_{programme}$, $LS_1\_programme\_director_{programme}$, $LS_2\_Teacher$, $LS_2\_lid_{Teacher}$, $LS_3\_Lecturer$, $LS_3\_tid_{Lecturer}\}$
f_2	$\{LS_2\_undergraduate\_programme$, $LS_2\_pid_{undergraduate\_programme}$, $LS_2\_year_{undergraduate\_programme}$, $LS_2\_phid_{undergraduate\_programme}$, $LS_3\_postgraduate\_programme$, $LS_3\_pid_{postgraduate\_programme}$, $LS_3\_year_{postgraduate\_programme}$, $LS_3\_phid_{postgraduate\_programme}\}$	$\{LS_2\_postgraduate\_programme$, $LS_2\_pid_{postgraduate\_programme}$, $LS_2\_year_{postgraduate\_programme}$, $LS_2\_phid_{postgraduate\_programme}\}$

Table 1. An Example

Suppose we have two quality criteria, c_1 and c_2. c_1 is the schema completeness quality criterion and c_2 is the mapping consistency criterion. c_1 is associated with a quality factor f_1, which defines schema completeness as the degree of coverage of local schema constructs that provide overlapping but possibly partially complete information for the same global schema constructs. c_2 is associated with quality factor f_2, which defines mapping consistency as the proportion of local schema constraints that are not violated by the new constraints introduced by the mappings. Table 1 lists the DI elements of the previous example that satisfy and do not satisfy quality factors f_1 and f_2.

Suppose now that there are three quality requirements issued by three users, A, B and C as listed in in Table 2. We discuss next the reasoning capability of our approach and how this can be used to determine the consistency of these requirements and the DI elements that violate any of them.

Given a quality hierarchy and logic statements representing different users' quality requirements, there are two validation steps in QFDI where

reasoning can be applied. First, reasoning can be applied in order to validate different users' requirements, as specified from their different quality perspectives. Second, reasoning can be applied in order to validate individual quality requirements with respect to the integrated resource. In the former case, inconsistent logic statements can be identified. In the latter case, the DI elements that do not satisfy individual logic statements can be discovered. When an inconsistency is discovered, the DI elements relating to quality factors referenced in the logic statements or the logic statements themselves may need to be modified in order to resolve such inconsistencies.

No.	**Requirement**	**Logic Statement in DL**
$A.1$	The set of DI elements that satisfy f_2 should be a subset of the set of DI elements that satisfy f_1	$\forall contains\_sat^{-}.\{f_2\} \sqsubseteq \forall contains\_sat^{-}.\{f_1\}$
$B.1$	The set of DI elements that do not satisfy f_1 should be a superset of the set of DI elements that do not satisfy f_2.	$\forall contains\_not\_sat^{-}.\{f_2\} \sqsubseteq \forall contains\_not\_sat^{-}.\{f_1\}$
$C.1$	The set of DI elements that satisfy f_1 should be disjoint from the set of DI elements that satisfy f_2.	$(\forall contains\_sat^{-}.\{f_1\} \sqcap \forall contains\_sat^{-}.\{f_2\}) \equiv \emptyset$

Table 2. Users' Requirements Example

The former case uses TBox reasoning and the latter case uses ABox reasoning [13]. In our example (see Tables 1 and 2), inferring from the users' logic statements first, without involving the DI elements (i.e., undertaking TBox reasoning), we can discover that $A.1$ and $C.1$ are not consistent since disjointness and subsumption cannot be satisfied by the same set of DI elements. Therefore, either $A.1$ or $C.1$ has to be modified. Suppose the data integrator removes $C.1$ and repeats the inference process. There is now no conflict between the remaining logic statements ($A.1$ and $B.1$). Next, for each remaining user requirement, we undertake reasoning again, this time including the DI elements (i.e., undertaking ABox reasoning). We can discover that A.1 is satisfied but B.1 is not since a subsumption relationship cannot exist between $contains\_not\_sat^{-}.\{f_2\}$ and $contains\_not\_sat^{-}.\{f_1\}$. Therefore, we know that the DI elements in these two sets need to be modified. In our example, suppose the data integrator modifies the mapping by removing the constraint $y > 1999$, and this makes $contains\_not\_sat^{-}.\{f_2\}$ empty. Both A.1 and B.1 are then satisfied.

4 Concluding remarks

We have presented a quality framework that is able to capture users' quality requirements in respect of integrated data resources. Our qual-

ity framework is formally represented in Description Logic, and users' quality requirements are expressed as logic statements over a variety of quality factors. This allows reasoning to be applied in order to discover inconsistencies between different user requirements using TBox reasoning. It also allows reasoning to be applied to validate the requirements using ABox reasoning, so that DI elements that do not satisfy them can be discovered. When an inconsistency is discovered, the DI elements relating to quality factors referenced in the logic statements or the logic statements themselves may be modified. The reasoning capabilities may then be re-applied in order to iteratively improve the quality of the integrated resource.

Our quality framework and reasoning process forms part of the DI architecture and methodology described in [3]. Our future work entails completing the implementation of that architecture and evaluating our approach and metrics with real-world case studies and users.

References

1. M. Jarke and Y. Vassiliou. Data warehouse quality: A review of the DWQ project. In *IQ*, pages 299–313, 1997.
2. S. Poslad and L. Zuo. An adaptive semantic framework to support multiple user viewpoints over multiple databases. In *Advances in Semantic Media Adaptation and Personalization*, pages 261–284. 2008.
3. J. Wang. A quality framework for data integration. http://www.dcs.bbk.ac.uk//research/techreps/2010/bbkcs-10-03.pdf, Tech.Rep. BBKCS-10-03, Birkbeck College, 2010.
4. M.B. Da Conceiao and A.C. Salgado. Information quality measurement in data integration schemas. In *Proc. QDB*, 2007.
5. A. Bonifati et al. Schema mapping verification: the SPICY way. In *Proc. EDBT*, pages 85–96, 2008.
6. K. Belhajjame et al. User feedback as a first class citizen in information integration systems. In *Proc. CIDR*, pages 175–183, 2011.
7. L.L. Yan et al. Data-driven understanding and refinement of schema mappings. *SIGMOD Rec.*, 30:485–496, May 2001.
8. L. Chiticariu and W. Tan. Debugging schema mappings with routes. In *Proc. VLDB*, pages 79–90, 2006.
9. A. Calì et al. Data integration under integrity constraints. *Inf. Syst.*, 29:147–163, April 2004.
10. L. Cabibbo. On keys, foreign keys and nullable attributes in relational mapping systems. In *Proc. EDBT*, pages 263–274, 2009.
11. R. Fagin et al. Data exchange: getting to the core. *ACM Trans. Database Syst.*, 30:174–210, March 2005.
12. J. Wang. Measuring quality in data integration settings. http://www.dcs.bbk.ac.uk/∼jianing/bbkcs-11-03.pdf, Tech.Rep. BBKCS-11-03, Birkbeck College, 2011.
13. F. Baader et al. *The description logic handbook: theory, implementation, and applications.* Cambridge University Press, 2003.