IADIS International Journal on Computer Science and Information Systems
Vol. 10, No. 2, pp. 32-51
ISSN: 1646-3692

DYNAMIC DATA QUALITY MANAGEMENT FRAMEWORK USING ISSUE TRACKING SYSTEMS

Mortaza S. Bargh. *Research and Documentation Center, Ministry of Security and Justice. The Hague, The Netherlands.*

Jan van Dijk. *Research and Documentation Center, Ministry of Security and Justice. The Hague, The Netherlands.*

Sunil Choenni1. *Research and Documentation Center, Ministry of Security and Justice. The Hague, The Netherlands. Creating 010, Rotterdam University of Applied Sciences. Rotterdam, The Netherlands.*

ABSTRACT

Organizations face many challenges in maintaining the quality of data in their information systems. Often offline methods like surveys are used in existing data quality management solutions. These methods, which are usually used in infrequent time intervals, suffer from high costs, high delays, and low fidelity inherently. In this contribution, we propose an innovative data quality management framework to dynamically monitor and improve the quality of data within an organization. To this end, the proposed framework relies on a problem resolving process, where users of information systems, e.g., data analysts, use issue tracking systems to report on data quality related problems, as these problems arise in post implementation phase of such information systems. Generally these reported problems are implicitly related to data quality issues. Thus, our proposed framework offers an automatic mechanism to semantically link these problems to data quality issues. Through this semantic linking, the framework offers added values for both data quality management community – who has traditionally relied on classic inquiries of human experts to detect data quality issues – and for data analyst community – who has traditionally relied on own expertise, thus rarely on state of the art data quality solutions, to resolve encountered problems. The paper discusses the set of functions included in the proposed data quality management framework and presents a proof of concept realization of the proposed framework.

KEYWORDS

Data Quality, Dynamic, Framework, Management, Problem resolving

1. INTRODUCTION

Organizations face many challenges for maintaining and managing their Information Systems (ISs) in the post-implementation phases of such systems. This issue can be characterized as operational changes and micro-changes (Lorenzi and Riley, 2000) that occur in organizations. Data Quality (DQ) management is an important part of IS management because quite often the datasets used in ISs are of low quality. As a typical example, we at the research center of Dutch Ministry of Security and Justice use the datasets of various departments of the ministry to produce insightful reports on judicial processes and crime trends for legislators, policymakers and the public (Braak et al., 2013; Choenni et al., 2010). Considering the diversity and distribution of our data sources, we often receive the corresponding datasets with missing, uncertain, inconsistent, etc. data records and attributes. Poor DQ may result in inaccurate and invalid data analysis outcomes, which can in turn mislead data consumers and end-users. Detecting and handling low quality data objects, i.e., as the main activities of DQ management (Berti-Équille, 2007), are challenging. For example, detecting the severity of DQ issues is a tedious operation carried out often through opinion surveys of data experts in infrequent intervals. This leads to late detection and thus management of DQ issues, which in turn causes undesired operational and strategic consequences for an organization.

Research and development in the field of DQ evolves along several directions, ranging from detecting DQ issues (Wong and Strong, 1996; Woodall et al. 2013; Pipino et al., 2002, EPA, 2006; Eppler and Wittig, 2000) to IS architecture improvements (Choenni et al., 2006; Verwer et al., 2013). Methods and techniques for detection of DQ issues start with identification of a set of the DQ attributes that are relevant for an organization. Often, questionnaires are used as a tool for identification of the relevant DQ attributes. After identification of relevant DQ attributes, the severity levels of DQ issues are determined and subsequently solutions are sought to adequately handle these DQ issues. Improving the IS architecture, on the other hand, is a long-term option and mostly requires that organizations make strategic decisions. A more practical approach would be dataset correction, which might statistically be meaningful, but its impact on DQ is not guarantied.

Seeking for an alternative solution direction, we note that data analysts in organizations often report on DQ related problems in Issue Tracking Systems (ITSs) such as wikis. As data analysts observe DQ related problems in their daily practice, they register these problems in ITSs in order to address them at appropriate times later on. For example, some typical problems that the data analysts in our organization have faced are: not being able to process criminal datasets on regional scale because the datasets were delivered at a national scale (Moolenaar et al., 2007), not being able to carry out trend analysis due to lack of historical criminal data records, or not being able to run concurrent queries because temporary datasets were distributed across various locations, where the latter is a problem also reported in (Birman, 2012). Our data analysts usually report such problems – although being related to DQ attributes of completeness and consistency – in terms of the observed symptoms and generic terms without linking them to DQ attributes. Subsequently, data analysts treat and address these problems based on their own expertise and experience. This isolative practice inflicts losses on organizations at operational and strategic levels. At an operational level, data analysts miss the rich state of the art solutions that can be found in DQ literature. At a strategic level, organizations and their management do not learn about existing DQ issues from the reported problems readily. The latter is a missing opportunity because managerial decisions are mostly made based on DQ issues.

In this contribution we aim at filling the gap between addressing the reported problems and managing the DQ attributes in organizations. Specifically, we present an innovative framework to measure and manage DQ attributes based on ITSs – where data analysts report on encountered (DQ related) problems and register the progress of treating those problems. Based on the proposed framework, we describe the design and realization of a DQ management proof of concept tool that can be used for maintaining DQ in databases and data warehouses. Conceptually, our proposed framework yields an automatic and dynamic DQ management that relies on user (i.e., data analyst) generated inputs. The resulting DQ management framework offers an added value for the DQ management community that has traditionally relied on classic surveying of human experts to learn about relevant DQ attributes. For a viable outcome such surveys used to rely on human experts, resulting in delays and high costs due to unavailability of such experts. The envisioned framework also offers benefits for data analysts involved in resolving reported (DQ related) problems through opening a realm of state of the art DQ solutions to them. The framework provides organization managers with a near real-time oversight on DQ issues so that they can base their managerial decisions on actual DQ issues. Hereby, DQ researchers, data analysts, and organization managers can benefit from the proposed DQ management framework.

We start with providing a background about DQ management as well as describing the motivation and the related work in Section 2. Subsequently we present the principles of our DQ management framework in Section 3 and the evaluation of the framework, including the implementation of the proof of concept tool, in Section 4. Finally we draw some conclusions and elaborate on future research in Section 5.

2. BACKGROUND

This section provides a background on the proposed DQ management framework by analyzing typical DQ management functionalities, the problem context and motivations of the work, and the related work.

2.1 Data Quality Management

DQ management is concerned with a number of business processes that ensure the integrity of an organization's data during its collection, aggregation, application, warehousing and analysis (AHIMA, 2012). As mentioned in (Knowledgent, 2014), DQ management “is the management of people, processes, technology, and data within an enterprise, with the objective of improving the measures of data quality most important to the organization. The ultimate goal of DQM” (DQ Management) “is not to improve data quality for the sake of having high-quality data, but to achieve the desired business outcomes that rely upon high-quality data.” Data quality can be characterized by a number of properties called DQ attributes. Like (Wand and Strong, 1996) we define DQ attributes as those properties that are related to the state of DQ.

DQ management consists of two main functional components of DQ assessment and DQ improvement, as shown in Figure 1. In the following we describe DQ management components in detail.

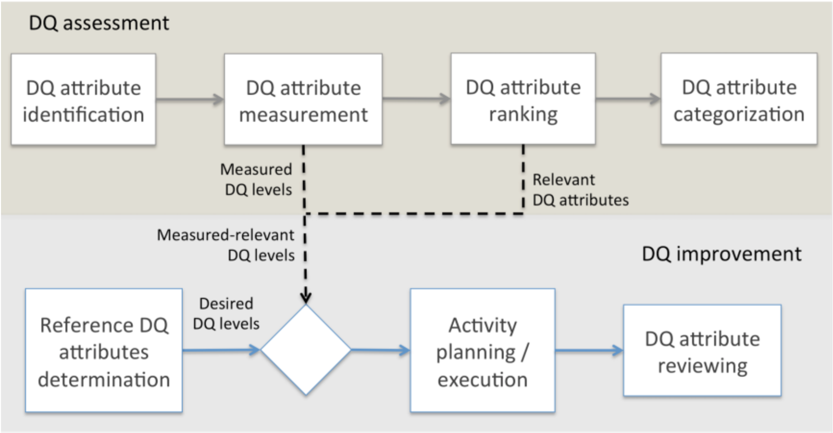

Figure 1. Functional components of DQ assessment (the top half) and DQ improvement (the bottom half).

2.1.1 Data Quality Assessment

DQ assessment is concerned with identifying, measuring, ranking, and categorizing of the DQ attributes that are relevant for an organization's data (Wang and Strong, 1996), as illustrated in Figure 1. One can characterize 'DQ attributes identification' by collecting possible DQ attributes from various sources like literature, data experts, and data analysts. 'DQ measuring' and 'DQ attribute ranking' are concerned with determining the importance of the identified attributes for the organization. 'DQ attribute categorization' is concerned with structuring the ranked attributes into a hierarchical representation based on the needs and requirements of the organization's stakeholders like data managers, data experts, data analysts, and data consumers (Wang and Strong, 1996).

2.1.2 Data Quality Improvement

DQ improvement is concerned with continuously examining the processes of data processing and enriching the quality of data in an organization that uses data as raw material. Figure 1 shows the functional components of DQ improvement, partly adopted from (Woodall et al, 2013). Given the relevant DQ attributes obtained from the DQ assessment, DQ improvement includes functionalities of 'reference DQ attribute determination' to identify the organization's requirements related to the related DQ attributes (i.e., the desired DQ levels, 'activity planning' to plan the solutions/activities for improving the relevant DQ attributes to the desired level, and 'data cleansing' to execute the planned data improvement activities. After applying the data cleansing activities, one needs also to do 'DQ attribute reviewing' in order to validate these activities based on their dependency, measure the DQ attribute levels, and to feed back the achieved results. Note that some consider the measurement of DQ attribute levels as part of DQ assessment, see for example (Woodall et al, 2013), therefore we have included this functionality in the DQ assessment as seen in Figure 1.

2.2 Motivation

ITSs are software products for managing and maintaining the lists of issues relevant for an organization. The tracked issues can be software bugs for software development houses (for which Bugzilla (2015) and JIRA (2015) are example ITSs), customer issues for customer support call-centers/helpdesks (for which H2desk (2015) is an example ITS), and assets for asset management companies (for which TOPdesk (2015) is an example ITS). ITSs are used by various stakeholders like software developers, customers, and employees of organizations for reporting on the issues they face. The reports include the (detailed) description of the problem being experienced, and information about issue urgency values (i.e., the overall importance of issues), who is experiencing the problem (e.g., external or internal customers), date of submission, attempted solutions or workarounds, a history of relevant changes, etc. ITSs were originated as small cards within a traditional wall mounted work planning. Therefore an issue report is also called tickets due to being a running report on a particular problem, its status, and other relevant data with a unique reference number. Organizations take appropriate actions to resolve them, based on these reports.

In our research center, the WODC (Bargh et al., 2015), there is an ITS for reporting and logging (DQ related) problems. As data analysts encounter problems in their daily practice, they write down these problems in this ITS. The goal of problem logging is to provide an overview of the existing problems to (other) practitioners like IT staff and data analysts, who shall resolve these problems based on their severity and urgency. Table 1 shows some typical problems registered in our ITS. Data analysts write done these problems in text (see the description column in Table 1) and for every problem they also provide two parameters: the momentary problem severity level and the desired problem severity level. The momentary problem severity level can be determined subjectively as perceived by the data analyst, or objectively as measured based on some data specific parameters. The data analyst determines the desired problem severity level subjectively. Both momentary and desired problem severity levels are expressed in a real number between 0 and 1, where 1 means the problem severity is the highest and 0 means the problem is resolved (i.e., the problem does not exist anymore and it can be removed from the ITS).

Table 1. 8 typical problems registered in our ITS and their descriptions.

Problem	**Description**
1	The column with community codes is missing in the table
2	The columns with community codes are missing in all tables
3	The column with community codes must be added
4	The column with community codes cannot be found in the table
5	The column with community codes is not filled
6	The columns with community codes are not filled
7	The community codes have been deleted

Figure 2 shows a typical problem resolving process in terms of its functional components, which is used for addressing the problems registered in ITSs. Data analysts – typically practitioners with a technical background in data science and databases – analyze the causes of a problem and (the impacts of) possible solutions in order to choose a solution based on some tradeoff criteria. After realization of the solutions some Key Performance Indicators (KPIs) are measured to derive the momentary problem severity level (i.e., to determine the effectiveness of the solution devised via the feedback loop). Although the registered problems are related to DQ attributes, their descriptions often do not include any direct link to the corresponding DQ attribute(s) because those data analysts who insert these descriptions have little knowledge about DQ concepts and definitions. Therefore, the textual definitions of problems are not specified in terms of DQ attributes. Not having a direct link to DQ attributes is a missing opportunity, which motivated us to integrate the depicted problem resolution process with DQ management so that both processes can benefit from the resulting framework.

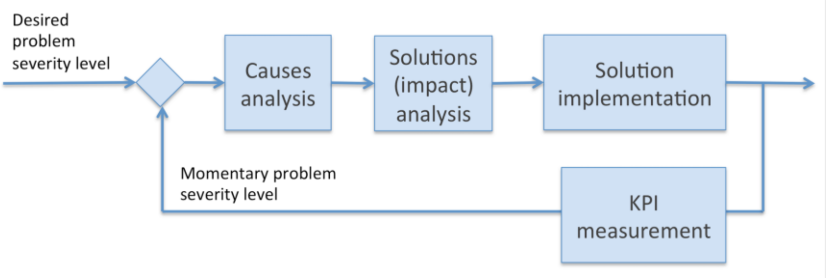
Desired problem severity level
Causes analysis
Solutions (impact) analysis
Solution implementation
Momentary problem severity level
KPI measurement

Figure 2. Process of resolving the DQ related problems registered at the ITS.

2.3 Related Work

A two-stage survey and a two-phase sorting method for identifying, ranking, and categorizing of DQ attributes in a given context are proposed in (Wang and Strong, 1996). The authors designed a survey to produce a list of potential DQ attributes by a group of participants of a workshop. Using another survey, the authors asked another group of the participants to rate the potential DQ attributes. An exploratory factor analysis of the importance ratings was applied to choose the most important DQ attributes. Finally, the authors applied a two-phase sorting method to categorize the chosen DQ attributes. For example, the DQ attribute categories mentioned in (Wang and Strong, 1996) are: Intrinsic, Contextual, Representational, and Accessibility. In most organizations, including ours, gathering so many participants, e.g., the data analysts in our organization, for surveying and sorting of DQ attributes is almost impossible due to time constraints or too few participants to produce valid results.

A so-called hybrid approach for DQ management is proposed in (Woodall et al., 2013). For a set of relevant DQ attributes, (Woodall et al., 2013) proposes to assess the required level of DQ improvement by comparing the current state to a reference state. The functional components of the hybrid approach are mentioned in Subsection 2.1.2. DQ management and improvement according to the hybrid approach remains very abstract because DQ issue diagnostics are made based on some high level strategic concepts. In our case, DQ management is intertwined with operational level practices of data analysts who observe and resolve (DQ related) problems. Establishing this link delivers a dynamic DQ management in our case, which is not the case in the hybrid approach.

All DQ assessment researches depend on some DQ objectives and try to find a set of relevant – also called targeted – DQ attributes based on those objectives. The Environmental Protection Agency (EPA) approach (EPA, 2006) relies on, among others, a review of DQ objectives, a preliminary review of potential problems/anomalies in datasets, and a statistical method to draw quantitative DQ related conclusions from the data. Our study uses also the idea of translating data problems into the DQ objectives, but by considering 'all' problems/anomalies reported in the data. This is unlike (APA, 2006) that considers just a few reported anomalies. Unlike (APA, 2006), we do not rely on statistical methods exclusively and, instead, we incorporate also the domain knowledge of data analysts.

Pipino et al. (2002) incorporate subjective DQ assessment in the EPA methods. To this end, the authors use a questionnaire to measure the stakeholders' perceptions on DQ attributes, for example, through inquiring the constraints of database administrators. Subsequently, the approach of (Pipino et al., 2002) determines the root causes of data discrepancies and tries to improve DQ issues by solving these discrepancies. Also our study combines both subjective and objective perceptions of the stakeholders on DQ related problems, but combining these perceptions is done at an operational level, i.e., based on the problems registered in the ITS, and, unlike (Pipino et al., 2002), not on a DQ attribute or strategic level. Consequently, we base our DQ improvement framework on an ITS. Also the approach of (Eppler and Wittig, 2000) for DQ management uses all EPA methods, but it adds some extra attributes to evaluate how pragmatic each DQ attribute is. The current study does not use any additional attribute to prove or determine how pragmatic the DQ attributes are.

ITSs are widely used for tracking and managing issues such as software bugs, customer issues, and assets that are relevant for an organization. While there are many applications of ISTs for collaborative software development, including also management of announcements, documentation and project website, there are no application of such systems for DQ management as we present in this contribution.

3. PROPOSED APPROACH

As shown in Figure 3, our proposed DQ management system architecture includes the functional components of a problem resolving process that serve as: (a) The DQ improvement part of the proposed DQ management system, see the lower part of Figure 3; and (b) The enabler of the DQ attribute ranking of the DQ assessment part, see the upper part of Figure 3. In the following, we shall describe those functional building blocks of Figure 3 that are specific to or are modified for our system (see those blocks marked with a *).

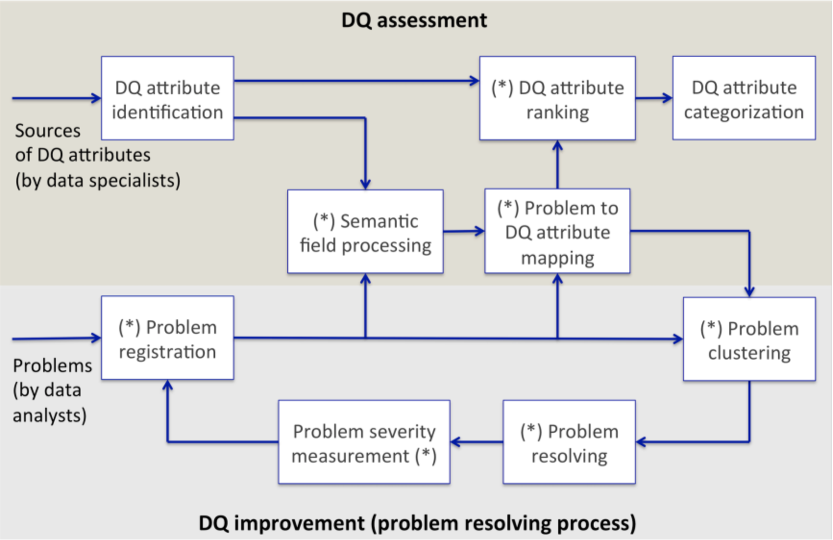
DQ assessment
DQ attribute identification
(*) DQ attribute ranking
DQ attribute categorization
Sources of DQ attributes (by data specialists)
(*) Semantic field processing
(*) Problem to DQ attribute mapping
(*) Problem registration
(*) Problem clustering
Problems (by data analysts)
Problem severity measurement (*)
(*) Problem resolving
DQ improvement (problem resolving process)

Figure 3. Functional architecture of the proposed DQ management framework.

3.1 Data Quality Assessment

The DQ attribute identification and categorization functionalities are not specific to our work, i.e., can be adopted from the literature, and therefore the corresponding blocks are not marked in Figure 3. Based on literature, we start with enlisting a number of potential DQ attributes and categorize the selected DQ attributes when they are ranked. In order to rank the identified DQ attributes, we rely on the set of the problems registered in the ITS. The registered problems are processed in the 'semantic field processing', 'problem to DQ attribute mapping', and 'DQ attribute ranking' components as described below.

3.1.1 Semantic Field Processing

A semantic field is a set of conceptually related terms (Kornai, 2010). In our setting, every semantic field corresponds to a DQ attribute. For any given DQ attribute the current study carries out two steps of:

a) Determining a list of the so-called 'related terms' that are related to the semantic field of the DQ attribute,
b) Syntactical decomposing of every related term to some phrases of smaller sizes.

In the first step, given a large list of potential DQ attributes obtained from the literature and given the actual problems descriptions registered in the ITS, every pair of (problem description, potential DQ attribute) must be analyzed. When a problem description is conceptually related to a DQ attribute, then the conceptual formulation of the problem description is recorded as a related term, which has a smaller size than the problem description size. Iteration of this step results in two columns of the 'related terms' and the corresponding 'DQ attributes' (or 'semantic fields') in a semantic field processing table, for an example see the two right columns in Table 2. In the second step, every related term is decomposed into sets of smaller phrases that syntactically appear in some problem descriptions. This results in another column in the semantic field processing table (i.e., see the left column in Table 2 with two sub-columns of phrase_1, phrase_2). Sometimes in this study, for simplicity reasons, we assume every related term contains at most two phrases.

Table 2. Example of a semantic field processing table (over the DQ attribute of 'completeness').

Phrase_1 *	**Phrase_2 ***	**Related terms**	**DQ attribute (semantic field)**
Is	Missed	Missing data	Completeness
Are	Missed	Missing data	Completeness
Be	Added	Adding data	Completeness
Not	Found	Missing data	Completeness
Is not	Filled	Empty fields	Completeness
Are not	Filled	Empty fields	Completeness
Is	Deleted	Lost data	Completeness
Are	Deleted	Lost data	Completeness

* Derived From problem descriptions

Note that a related term from the first stage is a natural language term. The syntactical decomposition of a natural language term can be ambiguous, so the same term can have more than one parsing tree (Mooney, 2007). For instance, 'missing data', can be syntactically decomposed to sets of phrase pairs of {Is, Missed}, {Are, Missed}, {Is, Missing} or {Are, Missing}. Domain experts define these semantic fields, related terms and phrases in a way that

- The phrases are found in problem descriptions of data analysts,
- Any phrase pair can be related to only one related term, and
- Any related term can be related to only one semantic field / DQ attribute.

For example, Table 2 shows a number of phrases that can be found in descriptions of user-defined problems and the corresponding semantic field / DQ attribute of 'completeness'.

The abovementioned guidelines induce a hierarchical, i.e., tree, structure on semantic fields, related terms, and phrase sets as illustrated in Figure 4. Due to this tree structure, there are no related terms that are common among semantic fields / DQ attributes, and there are no phrase sets that are common among related terms. If the linguistic terms of the phrase pairs are found in problem descriptions, one can map these problems to the corresponding related terms and DQ attributes. Based on the tree condition, every phrase set/pair can identify one related term and, in turn, one semantic field / DQ attribute. As any problem description may include more than one pair/set of phrases, however, the corresponding problem may be associated with more than one semantic field / DQ attribute as illustrated in Figure 4.

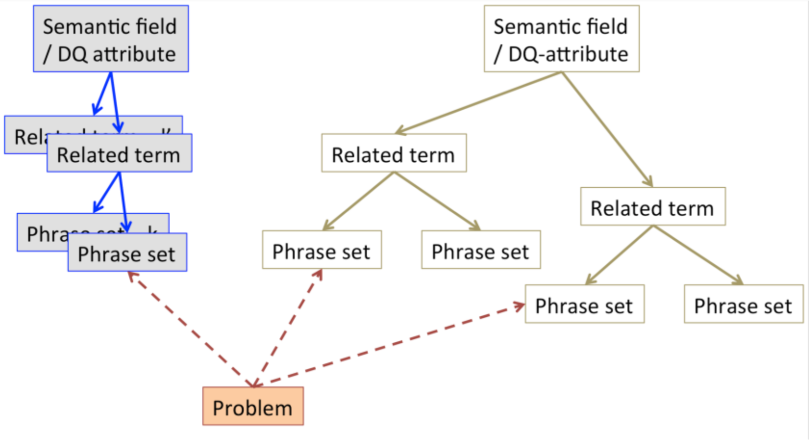

Figure 4. An illustration of the hierarchical structure of semantic fields, related terms and phrase sets; and the problems.

3.1.2 Problem to Data Quality Attribute Mapping

We define the problem to DQ attribute mapping as: For every problem registered in the ITS, determine the DQ attribute(s) with which the problem is associated. Given all phrase sets/pairs, we investigate the description of every problem registered in the ITS to seek out the phrase set(s)/pair(s) that appear in that problem description. Having found one or more phrase sets/pairs, we can associate the problem with the semantic field(s)/DQ attribute(s) corresponding to the related term(s) of the found phrase sets/pairs. Note that it is possible that some problems cannot be associated with any DQ attribute if the corresponding problem descriptions do not include any phrase set/pair identified in the semantic field processing table. Therefore, we aim at mapping as many registered problems as possible to DQ attributes, i.e., the number/ratio of such missed mappings should be zero ideally.

3.1.3 Data Quality Attribute Ranking

DQ attribute ranking is concerned with processing of the priority values of DQ attributes based on the number of the problems that are associated with these DQ attributes. For example, to determine the priority value of the DQ attribute ‘completeness’, we determine the ratio of the number of the problems mapped to the DQ attribute ‘completeness’ to the number of all problem to DQ attribute mappings.

For the DQ mapping mentioned above, we have weighed all mappings of problems to DQ attributes equally. Alternatively one can weigh these mappings differently based on some criteria or domain knowledge. For example, one can weight every mapping based on the difference between the momentary problem severity level and desired problem severity level, being a value in the unit interval of [0, 1]. We call this *weighed DQ attribute ranking*, which is dependent of the progress of the problem resolving process, i.e., the momentary problem severity level. As such the weighed DQ attribute mapping is *dynamic*. If we do not weigh the problem to DQ attribute mappings, then the DQ attribute ranking is only dependent of whether having a problem in the ITS or not. The underlying assumption is that a problem is removed from the ITS as soon as it is resolved. This DQ rank is called *static* because it does not change as the resolving of a problem progresses (unless it is totally resolved, thus removed from the ITS).

As an example, Table 3 summarizes the static ranks of the DQ attributes (see the middle column therein), obtained for the problems registered in our ITS. The DQ attributes are grouped in 4 categories of Intrinsic, Contextual, Representational, and Accessibility, as adopted from (Wang and Strong, 1996). These categories are ordered according to the sum of the corresponding DQ attribute ranks (see the second rightmost column). Within each category, the corresponding DQ attributes are ordered according to the DQ attribute ranks within that category (see the middle and the rightmost columns). Those DQ attributes with a ranking larger than zero are called targeted DQ attributes.

Table 3. An example list of ranked DQ attributes.

Category	DQ attribute	Overall rank	Category rank	Rank in category
1 – Contextual	1.1 – Consistency	0,251	0,767	0,327
	1.2 – Completeness	0,221		0,288
	1.3 – Validity	0,145		0,189
	1.4 – Relevancy	0,115		0,150
	1.5 – Volatility	0,025		0,033
	1.6 – Timeliness	0,010		0,013
2 – Representational	2.1 – Uniformity	0,074	0,161	0,460
	2.2 – Portability	0,024		0,149
	2.3 – Meta-documentation	0,021		0,130
	2.4 – Accuracy metadata	0,021		0,130
	2.5 – Compressibility-metadata	0,021		0,130
3 – Intrinsic	3.1 – Integrity	0,071	0,079	0,718
	3.2 – Fidelity	0,005		0,141
	3.3 – Variability	0,003		0,141
4 – Accessibility	4.1 – Data publication	0,056	0,078	0,460
	4.2 – Accessibility	0,011		0,149
	4.3 – Data connectivity	0,011		0,130

3.2 Data Quality Improvement

DQ improvement corresponds largely to the problem resolving process, as shown in Figure 3. Via solving problems registered in our ITS, data analysts also improve the corresponding DQ attributes, thus carry out DQ management. DQ improvement includes functional components of ‘problem clustering’, ‘problem resolving’, and ‘problem severity measurement’, as described in the following.

3.2.1 Problem Clustering

The aim of problem clustering is to reuse those solutions that address similar problems in the same cluster. Clustering brings optimization, efficiency, as it shortens the problem list. As explained in the previous section, the problem to DQ attribute mapping yields some (weighed) association values between pairs of (problem, DQ attribute). This (weighed) association values are input to the ‘problem clustering’ component. All problems that are similarly related to a set of targeted DQ attributes can be clustered in a set of problems. The clustered problems can further be classified according to some semantic criteria. The resulting clusters encompass those problems that share similar behaviors in terms of DQ attributes. In order to address registered problems one can prioritize these problem clusters, for example based on their sizes and weighs, and apply (and/or develop new) solutions that address these problem clusters according to their priorities.

Alternatively, one can *classify* problems in terms of existing solutions, instead of clustering them based on some behavioral similarity in the DQ attribute spaces. If, for example, there is a software tool that resolves/addresses a specific subset of DQ attributes well, then availability of such tool inspires us to consider classifying the registered problems in terms of the DQ attributes that are addressed by such a powerful software tool.

3.2.2 Problem Resolving

A set of activities must be planned in terms of, among others, the impact(s) of the corresponding problem(s) and the momentary/desired severity levels of those problems. DQ related problems impact the data environment by causing unwanted effects. When problem impacts are well modeled, in terms of for example their costs, one can prioritize the problems and start with solving most impactful/costly problems. Therefore, for example, we can classify problem impacts qualitatively based on four classes shown in Table 4 where problem impacts/costs decrease with the increment of the impact class's number. Knowing which problem to resolve first, one can apply appropriate solutions like those mentioned in Table 4 to contain the problem severity to the desired level.

Table 4. impacts classification and preventions of problems.

Impact class	Class description	Preventions/solutions
1	This problem incurs incorrect data, interruption of work, infinite accessibility, there will be a publication within 30 days, and fixing this problem is demanded.	For example, inform data analysts that DQ will be affected.
2	There are many projects involved, but the impacts of class one are not applicable.	For example, inform the leaders the affected projects.
3	There is only one project involved, but there are many data objects (tables, views, functions, procedures, packages, etc.) that are impacted.	For example, inform about affected objects.
4	Only one project and data object are involved.	For example: marking affected objects.

Previously we specified problems in the DQ attribute space by mapping problems to DQ attributes based on the severity and urgency of those problems. On the other hand, most solutions – including software tools and DQ improvement processes – can be characterized in terms of those DQ attribute issues that they resolve. Therefore, one can associate such solutions with the DQ attributes that they address. Moreover, one must balance the benefits of a solution against its costs. Various solutions inflict various costs on an organization. Therefore one can weight solutions based on their costs and accordingly weight the corresponding DQ attributes that those solutions address. Knowing the weights of DQ attributes both from their urgency and their cost, one can apply prioritization in the problem resolving process.

3.2.3 Problem Severity Measurement

KPIs are used to measure the momentary severity of problems in our current problem resolving process and they play an important role in our proposed DQ management framework. We, therefore, describe three methods that are currently used for KPI measurements: Subjective measurement, single point objective measurement, and multipoint objective measurement. In subjective measurement an expert, for example, the data analyst, assigns a problem severity level based on his/her insight and estimation of a problem severity at a given moment. In single point objective measurement the ratio of the number of low quality data objects to the number of total data objects is used. A data record can be an example data object. When a single point ratio is not enough to indicate the severity of a problem, a number of the last ratios could be used. We call this multipoint objective measurement. Figure 5-left shows two curves of per month single point measurements for two consecutive years. A multipoint objective measurement can be obtained from each of these curves by for example averaging monthly single point measurements or by averaging the difference between these two curves. The results of all three measurements is a real number between 0 and 1, which can be visualized by a Gauge or Dial chart as shown in Figure 5-right. The momentary severity level of a problem is updated in the IST as the problem's resolution process progresses.

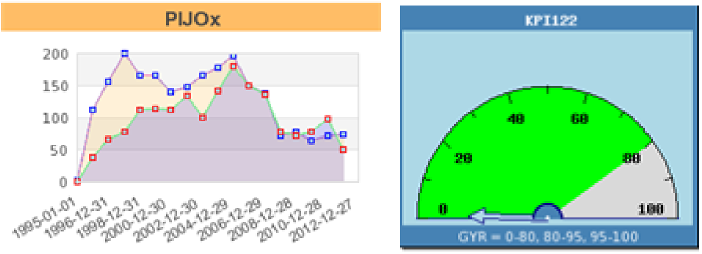

Figure 5. illustrations of two curves of multiple single point measurements (left) and dial chart graph (right)

4. EVALUATION

For the evaluation of the proposed DQ management framework we first review its functionalities compared to those proposed in the literature. Subsequently we describe the proof of concept prototype that is realized and deployed in our research center. For the realized prototype we further report on an evaluation of its 'semantic field processing' and 'problem to DQ attribute mapping' components.

4.1 Functionality

The proposed DQ management framework relies on the inputs of data consumers/analysts describing the DQ related problems they encounter in an organization. These user inputs are typically registered and updated in ITSs. We adopted the term framework here in its generic sense as used in software engineering. A software framework is a reusable environment that provides an abstraction about generic software functionalities, which can selectively be changed by additional application and domain specific software. According to (Eppler & Wittig, 2000), a DQ framework should achieve four objectives as described in the following, where we also elaborate upon the extent to which these objectives are realized in our proposed DQ management framework.

1. *Providing a systematic and concise set of criteria according to which [quality of] information can be evaluated*. We use the DQ related problems – as observed by data users in an organization – and map them to DQ attributes – using the 'sematic field processing' and 'problem to DQ attribute mapping' –to deliver a systematic and concise set of criteria for evaluating DQ observed in the organization.
2. *Providing a scheme to analyze and solve information quality problems*. We reuse those processes that organizations typically use to resolve the problems they encounter in practice. This is made possible by observing/reporting DQ related problems by data analysts and by a systematic mapping of these DQ related problems to DQ attributes.
3. *Providing the basis for information quality measurement and proactive management*. The mapping of DQ related problems to DQ attributes makes it also possible to use KPIs of observed problems for DQ attribute measurements.
4. *Providing the research community with a conceptual map that can be used to structure a variety of approaches, theories, and information quality related phenomena*. The proposed DQ management provides a generic model, i.e., a framework, for a (near) real-time management of DQ attributes. Each component can further be specialized for a given application and domain context. For example, the mapping of DQ related problems to relevant DQ attributes can be improved upon by using more detailed and domain specific models.

The DQ management framework presented in this contribution relies on a theoretical framework (Imenda, 2014) for DQ management, as summarized from DQ literature in Section 2. Based on a number of case studies, (Woodall et. al, 2013) identifies a number of activities that a DQ assessment approach includes. These activities are summarized in Table 5 where we also indicate with a * those data assessment activities that are recommended according to (Woodall et. al, 2013). The third column of Table 5 indicates how our proposed DQ management covers these DQ assessment activities.

Table 5. Positioning of the work with respect to the DQ assessment activities identified in (Woodall et al, 2013).

DQ assessment activities, copied/adopted from (Woodall et. al, 2013)		**As covered in our DQ assessment functionalities**
DQ activity	**Activity definition**	
* Conduct analysis of results	The process of analyzing the values from the DQ measurement(s)	In DQ attribute ranking and DQ categorization
Define DQ requirements	The process of defining what level of DQ is required. DQ requirements can be compared to the measurement values to determine required DQ improvement levels	Determining the desired level of problem severity by data analysts for observed problems
Expose the DQ assessment project to	Expose and establish senior management support for the DQ assessment project	Reporting the output of DQ categorization

senior management		
Communicate and share the results	Communicate and share the results or current progress of the DQ assessment with relevant people	
Group/organize data items	The process of grouping data items into categories (e.g., grouping criteria could be data type, level of risk)	
* Identify DQ dimensions	The process of identifying dimensions or using an existing model of DQ dimensions e.g., PSP/IQ DQ	DQ attribute identification
Identify and prioritize the organizational problems	Based on what is known at the start of the assessment, list the specific problems focusing on problems that relate to DQ	Use the desired and actual levels of problem severity and their urgency in DQ attribute ranking
Identify DQ costs	The process of determining the business impact and/or economic losses caused by low DQ (note that business impacts may not only be financial)	
* Identify DQ metrics	The process of identifying, developing or using an existing set of DQ metrics All	Problem severity measuring and semantically mapping the result to DQ attributes
* Perform objective/ subjective DQ measurement	The process of obtaining DQ measurements from an actual data set or by obtaining (subjective) opinions of the current state of DQ	Problem severity measuring using subjective and/or objective KPIs
* Select a place where data is to be measured	Select the place where data is to be measured based on the objectives for measurement. This includes determining when and where to measure the data or specifying, who will give subjective opinions.	
Model data creation and flow	The process of understanding and creating a model of the way data is created, updated, deleted and is transferred from one source to another	Problem resolving component
* Select data items	The process of selecting the relevant data values, attributes, tables, information systems, paper files etc. which will be subject to the DQ assessment.	These are done based on a natural process, as practitioners observe DQ related problems in their daily practice
Select processes	The process of selecting business processes that will be focused on in the Assessment	
* Identify reference data	The process of determining comparison data, which can be used as input to the selected metrics.	
Gather general meta data	The process of gathering relevant meta data such as data models DQA	Semantic field processing

Perform data profiling	The process of examining the data and collecting statistics and information about that data such as distribution of values	Problem severity measurement and mapping them to DQ attributes
Validate the DQ metrics	The process of checking that the DQ metrics and the implementation of DQ metrics are correct	Checking KPIs of problems in problem severity measurement

4.2 Proof of Concept

In order to enable data analysts to register the arising (DQ related) problems, we have used the Team Development environment of Oracle APEX as our ITS. The data log for this system is stored in an Oracle DBMS (Database Management System). Currently there are about 334 problems together with desired and momentary problem severity levels registered in the IST. The realized DQ management system includes finding the semantic fields that target the most of problems descriptions, mapping problems to DQ attributes, ranking the resulting DQ attributes based on the associations found, clustering of problems (currently based on a manual process), resolving problems in order of their impacts/costs, and measuring the momentary severity level of problems based on the described KPIs. The KPIs of single and multiple measurements are defined in SQL terms and visualized by a dynamic PHP website.

One of the main components of the proposed DQ management framework is ‘semantic field processing’. For this component we realized a heuristic as described below. Given a DQ attribute, the current implementation determines a list of the related terms for the semantic field corresponding to the DQ attribute, and semantically decomposes every related term to some phrase pairs of smaller sizes that syntactically appear in problem descriptions. Assume that we have a large number of potential DQ attributes, derived from for example the literature, and that we have the actual problems descriptions registered in the ITS. In the first step of the heuristic we analyze every pair of (problem description, potential DQ attribute). When a problem description is conceptually related to a DQ attribute, then the conceptual formulation of the problem description is recorded as a related term. This related term has a smaller size than the corresponding problem description size. Iteration of this step results in two columns of the ‘related terms’ and the corresponding ‘DQ attributes’ or ‘semantic fields’ in a semantic field processing table (see the two rightmost columns in Table 2). Lines (5) and (7) in the pseudo code below refer to this process. In the second step, every related term is decomposed into phrase pairs that syntactically appear in these problem descriptions. This results in another column in the semantic field processing table (see the leftmost column in Table 2, which consists of two sub-columns named phrase_1 and phrase_2). Lines (6) and (7) in the pseudo code below refer to this process.

```
     ⊳ SFP is set of rows of the semantic field processing table
     ⊳ rt is a related term
(1)  SFP ← ∅
(2)  for each problem description x do
(3)  for each potential DQ attribute dq do
(4)  if x refers to dq then
(5)      define rt as a conceptual formulation of dq
(6)      decompose x into (p1, p2)
(7)      if (p1, p2, rt, dq) ∉ SFP then SFP ← SFP ∪ (p1, p2, rt, dq)
```

For performance evaluation, we report here on the performance of the system component 'problem to DQ attribute mapping', which is directly a function of the performance of the 'semantic field processing' component sketched above. This component cannot target all problems in the ITS because we start with DQ attributes and look at the problem descriptions in the ITS to identify (a) the semantic field of every DQ attribute (i.e., the related terms) and (b) the set of (two) phrases per every related term. The latter (i.e., the phrases) are derived from problem description syntaxes. As a result, this process may overlook some problems, i.e., fail to map some of them to DQ attributes, due to not exhaustively searching the space of registered problems and DQ attributes. Our search of related terms and phrases stops at a certain point due to practical reasons, i.e., after finding a certain number of phrase-pairs.

Those problems that are (not) mapped to DQ attributes are called (un)targeted problems. In order to decrease the number of (un)targeted problems, one can further investigate all untargeted problems in an iterative process in order to come up with the (new) related terms corresponding to some (potential) DQ attributes. The number of untargeted problems decreases sharply with the number of related terms at the beginning of this iterative process, as shown in Figure 6. But at a certain point the number of untargeted problems does not decrease that much. This is because the descriptions of the remaining problems are poorly written and therefore we cannot associate them with any related term based on the syntax of the problem description. For example, problem description "X_J becomes negative" actually means that the integrity of the variable X_J has been violated, but that problem description cannot be retrieved by our problem to DQ attribute mapping because there is no phrase pair that matches with this description. One solution to map all problems is to add explicit and expressive key phrases to problem descriptions by data analysts during problem registration. This requires improving problem registration process, for example, by training and improving user awareness.

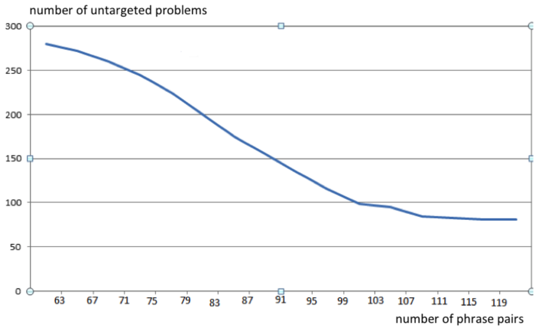

Figure 6. Number of untargeted problems (vertical) in terms of the number of related terms (horizontal).

4.3 Discussion and Limitations

In this study we proposed to measure the severity level of the reported problems (i.e., the KPIs) and map them to the corresponding DQ attribute levels. There are a number of challenges in measuring KPIs like determining/defining effective, valid, and standardized performance indicators. Making a KPI that measures the hamming distance of 2 words can be ineffective because, for example, the words "Netherlands" and "Holland" are semantically closer than their Hamming distances when considering their cultural background. In the proposed framework an underlying assumption is that data analysts in an organization register encountered problems in an ITS. The organization should encourage and train its employees to fill in the ITS so that the objectives of the proposed framework can also be fulfilled.

In this contribution we weighed the reported problems in the problem to DQ attribute mapping by the corresponding momentary and desired problem severity levels. One can think of other weighing methods based on problem and operational context. For example, also data analysts can weigh problems relatively when reporting them. In this way it is possible to distinguish between those problems that affect a small and/or less important dataset and those that impact multiple, large and important datasets.

Due to a prototype character of the current implementation, the ITS is deployed in another server and is loosely coupled to the rest components of the proposed framework. This hinders the communication between these two systems (as problem logs are downloaded as files currently). We intend to migrate the current implementation of the ITS and the rest of the DQ management system to have real-time communication among the components of the architecture.

5. CONCLUSION

In this contribution we used a problem resolving process, which relies on user generated inputs and ITSs, to design a dynamic DQ management framework. The resulting DQ management adapts to arising DQ related problems and the progress of resolving and addressing these problems. The proposed DQ management framework offers an added value for the DQ management community that has traditionally relied on classic surveying of human experts. For a viable outcome such surveys used to rely on human experts, resulting in delays and high costs due to unavailability of such experts in post implementation stages of ISs. The envisioned framework also offers benefits for data analysts involved in resolving DQ related problems through opening a realm of state of the art DQ solutions to them. In this way, both communities of DQ researchers and data analysts can benefit from one another.

A key component of this approach is a problem to DQ attribute mapping component that maps user generated inputs, i.e., the registered problems in ITSs by data analysts, to DQ attributes. The mapping provides a quantitative and dynamic means to determine the relevant DQ attributes and the level of their relevancy, given the operational setting, i.e., the desired and momentary problem severity levels as well as problem urgency level. We have realized the proposed framework as a prototype tool that operates according to the design objectives. We closely investigated the performance of the problem to DQ attribute mapping component and noticed that a fraction of problems become untargeted, i.e., we cannot map them to any DQ attribute. Through improving the problem registration process one can reduce the number of untargeted problems and guarantee their influence on DQ management process. It is for our future research to explore, for example, user awareness and training solutions. In the future, we intend o formalize the steps of the proposed DQ management framework and do research on improving, for example, automatic problem clustering, extending of objective KPI measurement, and improving problem impact analysis by incorporating weighs of the association between tuple (problem, DQ attribute) and considering the costs of available solutions.

ACKNOWLEDGEMENT

Preliminary results of this work are partially presented in (Bargh et al., 2015). In this contribution we have extended the work with an improvement of its architecture, the related work, and the evaluation section.

REFERENCES

AHIMA, 2012. Data Quality Management Model (Updated). *In Journal of American Health Information Management Association: AHIMA*. Vol. 83, No.7, pp. 62-67.

Bugzilla website, 2015. Retrieved on 30 Oct. 2015 from https://www.bugzilla.org.

EPA, Environmental Protection Agency, 2006. Data Quality Assessment: A Reviewer's Guide, *Technical Report EPA/240/B-06/002, EPA QA/G-9R*. Retrieved on 30 Oct. 2015 from http://www.epa.gov/QUALITY/qs-docs/g9r-final.pdf

Bargh, M.S., Choenni, S., and Meijer, R., 2015. Privacy and Information Sharing in a Judicial Setting: A Wicked Problem. *In Proceedings of the 16th Annual International Conference on Digital Government Research (DG.O)*, pp. 97-106, ACM.

Bargh, M.S., Mbgong, F., Dijk, J. van, and Choenni, S., 2015. A framework for Dynamic Data Quality Management. *In Proceedings of the 4th International Conference on Information Systems Post-implementation and Change Management (ISPCM),* July 21–23, Las Palmas, Grand Canary, Spain.

Berti-Équille, L., 2007. Quality Awareness for Managing and Mining Data. *Doctoral dissertation*, Université de Rennes 1.

Birman, K.P., 2012. Consistency in Distributed Systems. *In Guide to Reliable Distributed Systems*, book chapter, pp. 457-470.

Braak, S. van den, Choenni, R. and Verwer, S., 2013. Combining and Analyzing Judicial Databases. *In Discrimination and Privacy in the Information Society: Data Mining and Profiling in Large Databases*, Chapter 10, Springer, pp. 191-208.

Choenni, S. and Leertouwer, E., 2010. Public Safety Mashups to Support Policy Makers. *In Proceedings of Electronic Government and the Information Systems Perspective (EGOVIS)*, Bilbao, Spain, pp. 234-248, Springer.

Choenni, S., Blok, H.E., and Leertouwer, E., 2006. Handling Uncertainty and Ignorance in Databases: A Rule to Combine Dependent Data. *In Database Systems for Advanced Applications,* pp. 310-324, Springer.

Choenni, S., Dijk, J. van, and Leeuw F., 2010. Preserving Privacy Whilst Integrating Data: Applied to Criminal Justice. *In Information Polity, International Journal of Government & Democracy in the Information Age,* 15(1-2), pp. 125-138, IOS Press.

Eppler, M.J., and Wittig, D., 2000. Conceptualizing Information Quality: A Review of Information Quality Frameworks from the Last Ten Years. *In Proceedings of the Conference on Information Quality*, pp. 83-96.

H2desk website, 2015. Retrieved on 30 Oct. 2015 from: https://www.h2desk.com.

Imenda, S., 2014. Is There a Conceptual Difference between Theoretical and Conceptual Frameworks? *In Journal of Social Science*, 38(2), pp. 185-195.

JIRA software website, 2015. Retrieved on 30 Oct. 2015 from: https://www.atlassian.com/software/jira.

Knowledgent, 2014. White Paper Series, Building a Successful Data Quality Management Program. Retrieved on 30 Oct. 2015 from: http://knowledgent.com/wp-content/uploads/2014/09/Knowledgent-White-Paper-How-to-Build-a-Successful-DQM-Program.pdf

Kornai, A., 2010. The Algebra of Lexical Semantics. *In the Mathematics of Language*, pp. 174-199, Springer.

Lorenzi, N.M., and Riley, R.T., 2000. Managing Change: An Overview. *In Journal of the American Medical Informatics Association: JAMIA*, Vol. 7, No. 2, pp. 116–124.

Moolenaar, D., Choenni, S., and Leeuw, F., 2007. Design and Implementation of a Forecasting Tool for Justice Chains. *In Proceedings of the 5th Internations Conference on Law and Technology*, Berkeley, USA, September 24-26.

Mooney, R.J, 2007. Learning for Semantic Parsing. *In Computational Linguistics and Intelligent Text Processing*, Mexico City (invited paper), A. Gelbukh (Ed.), pp. 311-324, Springer.

Pipino, L.L., Lee, Y.W., and Wang, R.Y., 2002. Data Quality Assessment. *In Communications of the ACM*, Vol. 45, No. 4, pp. 211-218.

TOPdesk website, 2015. Retrieved on 30 Oct. 2015 from: http://www.topdesk.nl.

Verwer, S., Braak, S. van den, and Choenni, S., 2013. Sharing Confidential Data for Algorithm Development by Multiple Imputation. *In Proceedings of the 25th International Conference on Scientific and Statistical Database Management (SSDBM)*, pp. 42.

Wand, Y., and Wang, R.Y., 1996. Anchoring Data Quality Dimensions in Ontological Foundations. *In Communications of the ACM*, Vol. 39, No. 11, pp. 86-95.

Wang, R.Y. and Strong, D.M., 1996. Beyond Accuracy: What Data Quality Means to Data Consumers. *In Journal of Management Information Systems*. Vol. 12, No. 4, pp. 5-33.

Woodall, P, Borek, A., and Parlikad, A.K., 2013. Data Quality Assessment: The Hybrid Approach. *In Information & Management*, Vol. 50, No. 7, pp. 369-382.