Proceedings of the Federated Conference on Computer Science and Information Systems pp. 999–1003
DOI: 10.15439/2017F279
ISSN 2300-5963 ACSIS, Vol. 11

Domain-Specific Characteristics of Data Quality

Zane Bicevska
DIVI Grupa Ltd
Riga, Latvia
Email: Zane.Bicevska@di.lv

Janis Bicevskis
University of Latvia
Riga, Latvia
Email: Janis.Bicevskis@lu.lv

Ivo Oditis
DIVI Grupa Ltd
Riga, Latvia
Email: Ivo.Oditis@di.lv

Abstract–***The research discusses the issue how to describe data quality and what should be taken into account when developing an universal data quality management solution. The proposed approach is to create quality specifications for each kind of data objects and to make them executable. The specification can be executed step-by-step according to business process descriptions, ensuring the gradual accumulation of data in the database and data quality checking according to the specific use case. The described approach can be applied to check the completeness, accuracy, timeliness and consistency of accumulated data.***

Keywords- Data quality, domain-specific modelling languages, executable business processes

I. Introduction

The term "quality" depends highly on the context in which it is applied. The term is commonly used to indicate the superiority of a manufactured good or attest to a high degree of craftsmanship or artistry [1]. In manufacturing industries, quality is viewed as a desirable goal to be achieved through management of the production process.

Data quality is an IT-specific term, and it can be defined as the degree to which the data fulfills requirements of characteristics [2]. Examples of data quality characteristics are: completeness, validity, accuracy, consistency, availability, and timeliness.

The data quality problem is topical since over 50 years, and many different approaches are discussed in scientific publications addressing data quality issues. In the major part of sources the central attention is paid to defining of data quality characteristics informally and measuring of their values. Mechanisms for specifying of data quality characteristics in formalized languages usually are not considered. The main task of this research is to provide data quality management mechanisms being able to execute data quality specifications which are defined using formalized domain specific language (DSL).

To evaluate the data quality for the specific usage, the requirements for data must be described. The descriptions should be executable, as the stored data will be "scanned" and its' compliance to requirements will be checked.

In order to achieve the goal, two key requirements for specifying the data quality were formulated. Firstly, the ISO 9001:2015 standard considers data quality as a relative concept, largely dependent on specific requirements resulting from the data usage. It means the same data can be of good quality for one usage and completely unusable for another. For instance, to determine a count of students in a high school, only the status of students is of interest, not other data like students' age or gender. The same data may be checked for it's accordance to different quality requirements. It should also be emphasized that many conditions and requirements can't be checked during the data input as they are dependent on values of other data objects that are not entered yet. For instance, at the time of student's enrollment not all information about his/her financial obligations is available and/or entered in the database. This is the reason why high-quality data in practice occurs rarely.

The proposed approach intends creating of specific data quality model for each information system. The model is described by using means of a DSL, and it lets clearly define requirements for data objects attribute values and compatibility. The data quality model is executable: both the syntactic and the semantic controls are performed. The approach provides the possibility to use the data quality model for measurement of data quality.

The paper deals with following issues: overview about the related research (Section 2), and a description of the proposed solution (Section 3).

II. Related Works

There are three main research branches present: (1) the total data quality management (TDQM) theory, (2) the data quality defining by using the Object Constraint Language (OCL), (3) the data quality management using SSIS tools. They all are described in this chapter.

A. Total Data Quality anagement

The issue of data quality is essential since the very beginning of the IT industry. Numerous studies have led to variuous definitions of data quality. For instance, data are of good quality if they satisfy the requirements imposed by the intended use [3].

The research leading to these results has received funding from the research project "Information and Communication Technology Competence Center" of EU Structural funds, contract nr. 1.2.1.1/16/A/007 signed between ICT Competence Centre and CFLA of Latvia, Research No. 1.8 „Data Quality Management by using Executable Business Process Models".

Data are of high quality if they are fit for their intended uses in operations, decision making, and planning. According to Joseph Juran [4] data are fit for use if they are free of defects (accessible, accurate, timely, complete) and possess desired properties (relevant, comprehensive, proper level of detail, easy to read, easy to interpret) [5].

Data quality can also be characterized by different dimensions. In 1996 Wang and Strong [6] defined 15 data quality dimensions which are confederated in four quality groups: intrinsic, contextual, representational, accessibility.

Redman [5] provides 51 data quality dimensions, arranged in 9 data quality groups. Such a deep data quality gradation may seem an overstatement, especially for practitioners. In 2013 the Data Management Association International UK Working Group possesses only 6 dimensions: Completeness, Uniqueness, Timeliness, Validity, Accuracy, Consistency.

B. Object Constraint Language

The OCL started as a complement of the UML notation with the goal to overcome the limitations of UML in terms of precisely specifying detailed aspects of a system design [7]. Since then, OCL has become a key component of any model-driven engineering (MDE) technique as the default language for expressing all kinds of (meta)model query, manipulation and specification requirements [8].

Constraints at the model level state conditions that the "data" of the system must satisfy at runtime. Therefore, the implementation of a system must guarantee that all operations that modify the system state will leave the data in a consistent state (a state that evaluates to true all model invariants). Clearly, the best way to achieve this goal is by providing code-generation techniques that take the OCL constraints and produce the appropriate checking code in the target platform where the system is going to be executed.

Typically, OCL expressions are translated into code either as database triggers or as part of the method bodies in the classes corresponding to the constraint context types. Roughly, in the database strategy each invariant is translated as a SQL SELECT expression that returns a value if the data does not satisfy that given constraint. This SELECT expression is called inside the body of a trigger so that if the SELECT returns a non-empty value then the trigger raises an exception. Triggers are fired after every change on the data to make sure that the system is always in a consistent state. The OCL has many positive qualities: (1) OCL is an extension of UML, and it has gained a wide popularity in the computer scientists' community, (2) OCL provides a rich range of means of expression, allowing the use of widely used programming constructions. At the same time the disadvantages of OCL should also be recognized: (1) OCL is a declarative language without graphical notation, (2) contraints of OCL are closely related with the data storage in a relational database, (3) defining of data quality constraints in OCL requires good programming skills.

Furthermore, the OCL is missing a number of features that are necessary for data quality:

- no data read/write operations,
- no operations for reading and checking of discrete data objects that are not related to database (such operations are necessary for verifying of data entered via screen forms),
- constraints in OCL are described linearly (like a program code) and not graphically,
- defining and understanding of OCL constraints requires deep knowledge and skills in object-oriented programming; it makes the OCL unsuitable for industry professionals without appropriate IT background.

Usually data quality controls are hard-coded in data processing programs and can not be changed without involvement of programmers. As a result, often inconsistent data is entered and stored in databases.

OCL-based data quality solutions are hard to use practically due to the dynamic data input into database as well as to the complexity of OCL.

C. SQL Server Integration Services

As every solution, Microsoft SQL Server Integration Services (SSIS) has various advantages and disadvantages [9]. SSIS offers wide range of features for data migration, and designing of ETL and transformation processes [10]. To cover a broad spectrum of requirements for data migration and ETL processes, SSIS includes both standardized operations for many widely-used database management systems, and add-ons for different import/ export formats, and opportunities for developers to use the programming environment VisualStudio.

Furthermore, SSIS is open platform allowing create and use external add-ons. Hence SSIS should be considered as a mature platform that is suitable not only for solving of ETL tasks but also for processing of emails, linear text files, XML files, and other oprations. The rich range of included features enables creating of SSIS packages from predefined components or to develop them by programming.

Microsoft has designed this product to provide better approach towards data migration, manipulation and transformation. With the power to define the workflow of process and task, user can easily define how the process should flow and perform some task on different interval. It also provides color codification and real-time monitoring.

SSIS advantages:

- SSIS can handle data from heterogeneous data sources,
- SSIS provides transformation functionality,
- Tightly integrated with Microsoft Visual Studio and Microsoft SQL Server,
- suitable for complex transformations, multi-step operations and structured exception handling.

SSIS disadvantages:

- to see package execution report needs Management Studio rather than being published to reporting services,
- SSIS memory usage is high and it conflicts with SQL.

Authors of [11] assure that usage of SSIS removes need of hardcore programmers as SSIS is apparently easy to understand and manage. In contradiction to [11] the authors of this research believe that the usage of SSIS have some fundamental barriers. The complexity of the approach is high; the usage of the solution for data processing and data quality management require either programmer's level of understanding of process execution, or many years of experience with SSIS.

Although not designed specifically for data quality management, features offered by SSIS provide a number of suitable solutions. Currently there are not known SSIS uses for data quality management which were not related to data migration. However, data quality management elements offered by SSIS are practically usable and should be taken over in further data quality solutions.

III. PROPOSED APPROACH

The data is stored in the database gradually in various steps. Hence the data quality requirements should be formulated for several levels of a data object – (1) discrete data object, (2) contextual control on interrrelated data, (3) contextual control on the database, and (4) contextual control on several databases.

A. Data Quality Requirements for a Separate Data Object

The proposed ideas will be demonstrated with the help of a simple example. Let us consider a working time tracking (WTT) system having the ER model given in the Fig.1. There are many active projects in an enterprise (entity Projects); every project has several employees (entity Developers); each employee (developer) may be involved in several projects; the working time spent by an employee (developer) in a specific time frame is aligned to one specific project (entity Work_time).

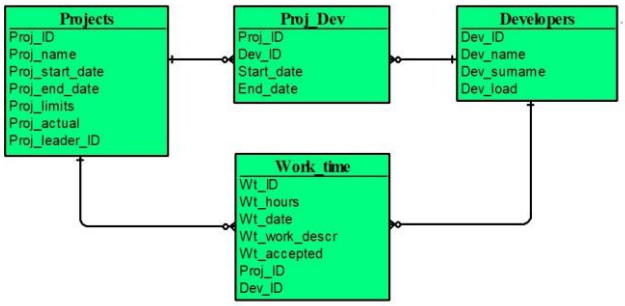

Fig. 1 WTT data model

The entity *Projects* has the following attributes: *Proj_ID* (project identifier to specify the project to which the spent working time should be referred), *Proj_name* (project name), *Proj_volume* (the estimated work amount of the project in man-hours), *Proj_start_date, Proj_end_date, Proj_limits* (the maximum allowable work amount of the project in man-hours), *Proj_actual* (project is active/passive), *Proj_leader_ID* (project manager).

The entity *Developers* has the following attributes: *Dev_ID* (the developer to whom the spent time should be referred), *Dev_name* (developer's name), *Dev_surname*, *Dev_load* (the minimum monthly developer's workload).

The entity *Work_time* has the following attributes: *Wt_ID* (identifier of the spent working time record), *Wt_hours* (spent working time of the developer), *Wt_date* (date of the spent working time), *Wt_work_descr* (description of the performed work), *Wt_accept* (reported working time is accepted by the project manager, Yes/ No), *Proj_ID* (the project to which the time should be referred), *Dev_ID* (the developer to whom the time should be referred).

The entity *Proj_Dev_time* is a junction table for dealing with many-to-many relationships, and it has the following attributes: *Proj_ID* (the project where the Dev_ID works), *Dev_ID* (the developer working in the project Proj_ID), *Start_date* (the date when the developer Dev_ID started to work in the project Proj_ID), *End_date* (the date by which the Dev_ID will be assigned to the Proj_ID).

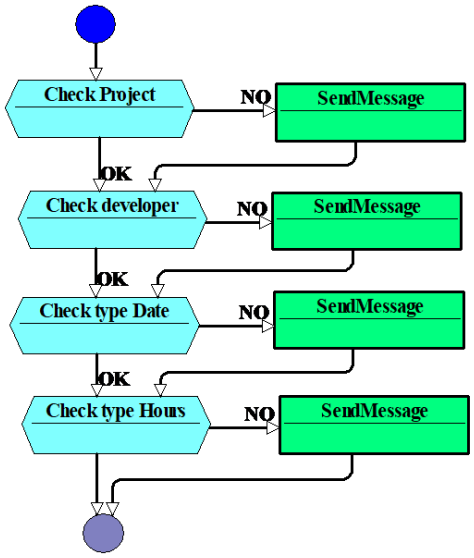

Fig. 2 Example of data object's syntactic control

Let us assume, the developers prepare reports about their working time autonomously and send the reports to data base where all enterprise data from various sources is collected. The procedure receives values of attributes:

< *Proj_ID, Dev_ID, Wt_date, Wt_hours, Wt_work_descr* >

The quality specification of report shown in the Fig.2 ensures quality control within one input message: (1) are all mandatory fields completed (Proj_ID, Dev_ID)?, (2) have input values correct data types (Wt_date, Wt_hours)?

In order to make the quality specification executable, informal texts should be replaced by program routines executing the desired operations.

B. Contextual control on interrelated data

Contextual control on interrelated data (see Fig.3) ensures quality control using attribute values of mutually interconnected data objects: (1) does the message contain object instances with references to other data objects (Project exists, Developer exists)?, (2) are the attribute values of input data in compliance with related data objects?

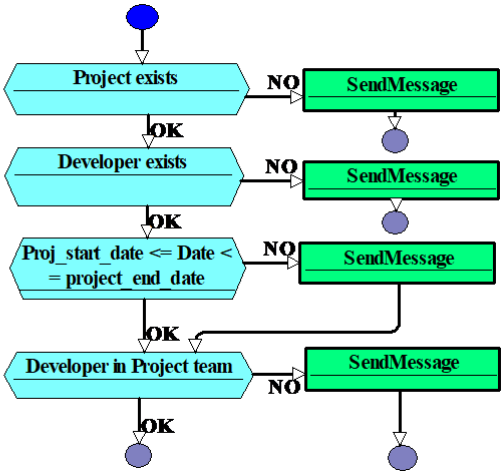

Fig. 3 Example of contextual control on interrelated data

In order to make the quality specification executable, informal texts should be replaced by SQL statements for data retrieving and control of constraints (see. Fig.4). Tools like SSIS may be used – these also offer statements for execution of SQL statements and validation of results.

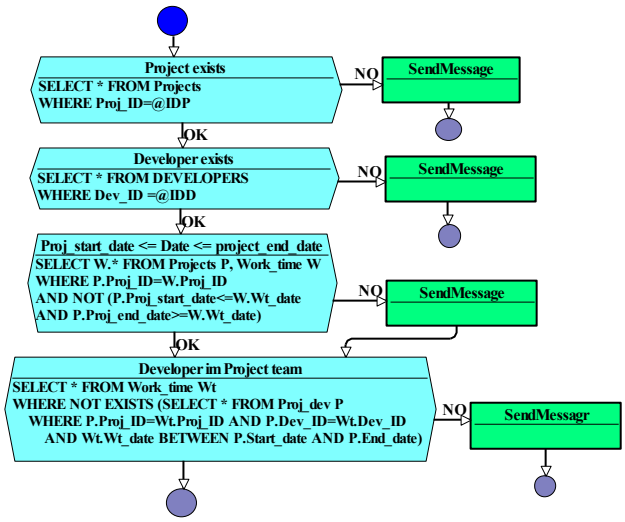

Fig. 4 Example of executable contextual control on interrelated data

C. Contextual control on the database

Contextual control on the database (see Fig.5) checks the compliance with conditions valid for the whole data base (examples: isn't the maximum of work amount allowed for the project exceeded, do the reports of employee cover the minimal workload of the employee in the time period, etc.).

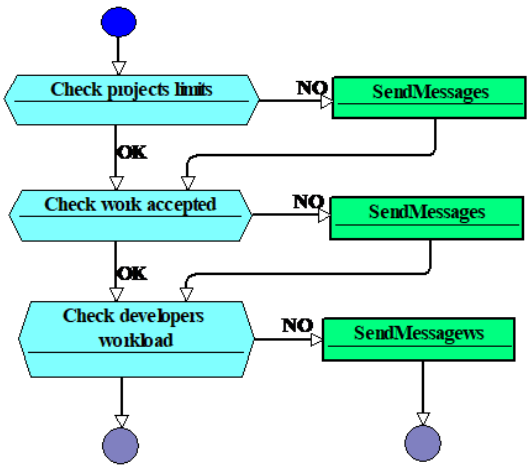

Fig. 5 Example of contextual control on the database

In order to make the quality specification executable, informal texts should be replaced by SQL statements for data retrieving and control of contraints (see. Fig. 6).

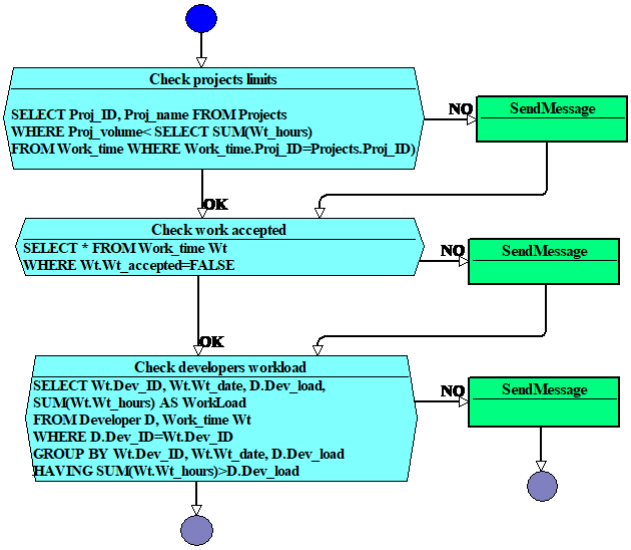

Fig. 6 Example of executable contextual control on the database

Diagrams in Fig.2, Fig.3 and Fig.5 form an informal data quality specification of working time tracking (WTT) system. It may be useful for industry experts to describe data quality requirements as a more formalized specification of executable controls is not practically applicable without IT skills. Executable data quality specifications can be used in business process steps to check data quality in certain points of processes.

Practical uses of the proposed approach have shown advantages of graphically represented data quality specification as they were more effective in discovering of information system-caused data quality errors than the traditionaly used informal data quality specifications in the textual form. It reaffirmed advantages of graphic diagrams in comparision to natural language texts in standardized documents.

Additional advantages can be gained if the data quality specifications are transformed to executable specifications. Although additional programming is needed to ensure the executability, a much higher dara quality can be achieved in an information system as a whole if data quality controls are incorporated in business process steps.

First two of mentioned data controls tipically are applied during data input. In case data should be saved anyway it is marked as incorrect. As this one control over database could be rather resource-intensive (time, server memory, processor time, data locking) it can not be executed on every data manipulation. Contextual control on the database usually is executed out of business hours and even not every day. Still the proposed approach is universal, and it is applicable in different cases – during the initial data input in information system, migrating data from one information system to another, performing data transformation to data warehouse.

D. Contextual control on several databases

The above described approach is also applicable in cases when several information systems of different enteprises are involved. Such a case is typical for public institutions with different but interrelating state information systems.

This problem has been addressed in Latvia since 2000. The essential data of public interest are accumulated in different state information systems: Population Register, Business Register, Vehicle register, etc. Each of the registers is managed and maintained by some public body which is responsible for the quality of the accumulated data.

The registers should also mutually exchange data; usually it is organized with the help of web services serving and receiving data – concrete values of data objects' attributes. Each data exchange session may require only few attribute values. When using data quality specifications, it is possible to check and evaluate the quality of received data.

Like the Latvian Integrated State Information System project [12], the described problem is also addressed in Estonia [13], Lithuania [14] etc.

Currently development of various industry-specific state information systems is continuing, and the identified data quality problem persists in each system again and again.

IV. CONCLUSIONS

The research shows the relative and dynamic nature of data quality. The usage of data implies requirements for data quality; the data are accumulated and verified step-by-step. Consequently a data quality management system has to fulfill the following key requirements:

- Data quality requirements are formulated for different levels – a discrete data object, interrrelated data objects, data in a database, data in several databases.
- Data quality requirements should be specified in an easy-to-understand definition language to ensure that industry experts will be able to formulate data quality requirements without involvement of IT professionals. It is advisable to use graphical DSL.
- If considering usage of the OCL – a quite popular language in computer scientists' community - , there should be taken into account that OCL is declarative language without graphical notation, it's constraints are closely related to the way how data is stored in a relational database, and formulating of data quality constraints is rather complicated and requires good skills in programming.
- The SSIS, developed by Microsoft, offers a range of useful features for data quality management including extracting data from different types of information sources and checking the validity and correctness of different data object relations. SSIS is an integrated part of Microsoft SQL Server and Visual Studio.

The proposed approach and tools for designing of executable data quality specifications in different levels let to design, develop and use the specifications as steps in executable business processes.

The paper is a continuation of authors' researches in the area of executable models and DSL [15], [16], [17].

REFERENCES

[1] Veregin H. Data quality parameters. In P A Longley, M F Goodchild, D J Maguire, and D W Rhind (Eds.) New Developments in Geographical Information Systems: Principles, Techniques, Management and Applications, John Wiley & Sons, Inc. (2005), pp. 177-189
[2] ISO 9001:2015. Quality management principles https://www.iso.org/standard/62085.html
[3] Olson J.E. Data Quality. The Accuracy dimension. Morgan Kaufmann Publishers (2003), p. 294
[4] Juran J.M., Gryna F.M. Juran's quality control handbook, 4th ed. New York: McGraw-Hill (1988)
[5] Redman T.C. Data Quality. The Field Guide, Digital Press (2001), p. 74
[6] Wang R.Y., Strong D.M. Beyond Accuracy: What Data Quality Means to Data Consumers, Jurnal of Management Information Systems, Springer, Vol.12., No.4 (1996), pp. 5-34.
[7] OCL 2.0. Object Constraint Language™, Version 2.0. Release date: May 2006. http://www.omg.org/spec/OCL/2.0/
[8] http://www.omg.org/spec/OCL/2.4
[9].https//www.codeproject.com/Articles/155829/SQL-Server-Integration-Services-SSIS-Part-Basics
[10] Features Supported by the Editions of SQL Server 2014. msdn.microsoft.com. Microsoft Developer Network..
[11] Sarjen, Microsoft Practices. What is SSIS? Its advantages and disadvantages. http://www.sarjen.com/ssis-advantages-disadvantages/
[12] http://www.varam.gov.lv/eng/darbibas_veidi/e_gov/?doc=13052
[13] https://www.ria.ee/en/administration-system-of-the-state-information-system.html
[14] https://ivpk.lrv.lt/en/activities/state-registers-and-information-systems
[15] J.Bicevskis, Z.Bicevska, Business Process Models and Information System Usability, Procedia Computer Science 77 (2015), 72 – 79.
[16] J.Ceriņa - Bērziņa, J.Bičevskis, Ģ.Karnītis "Information systems development based on visual Domain Specific Language BiLingva", In: 4th IFIP TC2 Central and East European Conference on Software Engineering Techniques (CEE-SET 2009), Krakow, Poland (2009)
[17] Bicevska, Z, Bicevskis, J, Karnitis, G. Models of event driven systems. Communications in Computer and Information Science Volume 615, 2016, Pages 83-98