Intelligent Information Management, 2011, 3, 230-239
doi:10.4236/iim.2011.36028 Published Online November 2011 (http://www.SciRP.org/journal/iim)

A Rule Management System for Knowledge Based Data Cleaning

Louardi Bradji, Mahmoud Boufaida
LIRE Laboratory, *Mentouri University of Constantine Ain El Bey*, *Constantine*, *Algeria*
E-mail: *bradjilouardi@yahoo.fr*, *mboufaida@umc.edu.dz*

Received July 23, 2011; *revised September* 12, 2011; *accepted September* 22, 2011

Abstract

In this paper, we propose a rule management system for data cleaning that is based on knowledge. This system combines features of both rule based systems and rule based data cleaning frameworks. The important advantages of our system are threefold. First, it aims at proposing a strong and unified rule form based on first order structure that permits the representation and management of all the types of rules and their quality via some characteristics. Second, it leads to increase the quality of rules which conditions the quality of data cleaning. Third, it uses an appropriate knowledge acquisition process, which is the weakest task in the current rule and knowledge based systems. As several research works have shown that data cleaning is rather driven by domain knowledge than by data, we have identified and analyzed the properties that distinguish knowledge and rules from data for better determining the most components of the proposed system. In order to illustrate our system, we also present a first experiment with a case study at health sector where we demonstrate how the system is useful for the improvement of data quality. The autonomy, extensibility and platform-independency of the proposed rule management system facilitate its incorporation in any system that is interested in data quality management.

Keywords: Rule, Data Quality, Data Cleaning, Knowledge, Rule Management System, Rule Based System, Structure

1. Introduction

Data quality (DQ) has always been an important issue and is even more the case today. The research works look at the role of Data Cleaning (DC) tools in helping improve DQ and clarify the need to take an enterprise-wide approach to DQ management, which is increasingly complex, open and dynamic [1,2]. Although there is a steady stream of DC tools in practice and research, more recent studies have showed that the situation has not enhanced and companies do not invest adequately in DQ [3]. Therefore new approaches and architectures are required to help improve the DQ.

There is a wide variety of DC tools. Their functionality can be classified as follows: Declarative DC and Rule-Based approaches for DC (RBDC). Whatever tools are chosen, a systematic, the rule-based approach yields better results than an unstructured approach [4-6]. Although the Rule Based Systems (RBS) that encode knowledge as rules and used to process complicated tasks have been firmly established for many years, they have not been well formally and adequately addressed for the DC tasks. Hence there is a need to an appropriate RBS for DC.

The current RBDC solutions often use Business Rules (BR), which are constraints on data contained in business policies into a formal form in order to allow computerization [7]. As BR are only a specific part of all Domain Knowledge (DK), it is necessary to investigate, how to exploit the all DK for DC. The main challenge in DK is knowledge discovering, which is usually manually made by human experts (interviews or questionnaires) or collected from written sources (books or other reference material). Consequently, the first step in the development of any RBS for DC is to begin with a collection of the knowledge from which the rules will be derived. Organizing the collected knowledge so that translation to rules will be straightforward is also a challenging task for the knowledge engineer [2,8].

Several research works have shown the importance of

DK in DC. However, there are few works on knowledge management issues for DC.

Similar to data, knowledge are also issued from different sources, which may be incomplete, inconsistent, heterogeneous and pervaded with uncertainty. Then, there is a need and a wide-ranging interest in managing the quality of knowledge and rules in order to ensure high DC quality.

In this paper, we assume that a RBS can play yet another role; it can be an agent of change to improve the DQ. Our primary objective is to improve the functional capability of a DQ management by embedding it with knowledge of the problem area. When the knowledge acquisition is performed, the system states the knowledge in the form of rules and uses these embedded rules to provide advice. As the currently RBS allow only the manipulation of production rules without precise understanding of rules theoretical foundations and without considering rule quality, our system provides an uniform rule representation based on First Order Structure (FOS) theory of Logic that allows the representation of all the types of rules (derivation, production, integrity, transformation and reaction) and their quality characteristics. Hence the proposed RBS for DC contains two subsystems: the first is related to the rules management and the second is focused on the rules quality management. In this system, we have incorporated two semi-automatic processes for the knowledge acquisition and transformation.

Another feature of our proposal is that the control and the improvement of rule (resp. knowledge) quality are performed on-line, incrementally, during the design of the system. It permits an early clean of knowledge and rules from anomalies and inconsistencies.

The proposed RBS for DC system lends itself to iterative process because in the first time it has not all the information needed to write the knowledge and rules.

We assume that our proposal can be very advantageous in the meaning that the RBS can interject and provide necessary knowledge at the appropriate points in the DC process.

The rest of this paper is organized as follows: In Section 2 the concepts of KBS and RBS, and their applications are given. It also provides some related works of RBDC followed by brief discussion for the motivation of this research axis. Section 3 gives an overview of the architecture of our system RBS-based DC. We formalize in this section the unified rule representation based on the use of FOS theory in order to represent all types of rule and its quality characteristics. In section 4, we give some details of the rule management components. The fundamental functions of the system are described in Section 5. In Section 6, a case study and some experimental results of applying the system to a health sector can be found. Finally, a conclusion and some perspectives are drawn in Section 7.

2. Background and Some Related Works

As our objective is to enhance the DQ by applying a Rule based approach in DC, it is necessary then to represent some works related to Knowledge Based System, Rule Based System and Rules-Based Data Cleaning.

2.1. Knowledge Based Systems

A *KBS* is defined as an interactive software system that uses stored knowledge of a specific problem to assist and to provide support for decision-making activities bound to the specific problem context. KBSs have been developed and used for a variety of applications [9-12].

Knowledge acquisition is the most important element in the development of KBS. Knowledge could be obtained from domain experts, raw data, documents, personal knowledge, business models and/or learning by experience [12,13].

The KBS is vague, and that it is generally hard to specify in advance a KBS completely because of the incremental nature of the knowledge elicitation process [14].

Interviews are considered as the most popular and widely used form of expert knowledge acquisition. One serious drawback of interviews is the fact that they are time-consuming [15].

A review of the literature on knowledge systems highlights the following major many advantages [16]:

- Quality improvement.
- Practical knowledge made applicable: Systems can assist experts in decision making even if they have that knowledge to hand; this improves the accuracy and timeliness of the decision made.
- Infallible and complete.
- Consistency: Results produced by a knowled- ge system are consistent throughout its operational lifespan.
- Updating knowledge.

Very often people express knowledge as natural language or using letters or symbolic terms. There exist several methods to extract human knowledge [13].

2.2. Rule Based Systems

*RBS*s prove to constitute one of the most substantial technologies in the area of applied Artificial Intelligence (AI). They have used in diverse intelligent systems. Some interesting recent applications of RBS are the ones that are related to business and the Semantic Web. Both current research works and applications go far beyond the traditional approach [13,17].

In the RBS, knowledge is represented in the form of (If condition Then conclusion < action >) production rules [18,19]. A rule describes the action that should be taken if a rule is violated. The empirical association between conditions and conclusions in the rule base is their main characteristic. The general architecture of RBS includes domain independent components such as the rule representation, the inference engine, the explanation system, a rule engineering tool and a specific user interface [15, 20]. Modern shells for the development of RBS, such as CLIPS, Jess, Drools, Aion, Ilog Rules, or Gensym's G2, follow this architecture. Most of this RBS are just shells providing a rule interpreter and sometimes an editor [21].

Various rule representation techniques are available. However, the rules for DC would require an appropriate rule representation technique that allows the management of rule and its quality, and the presentation of all type of rule [20].

Let us notice that there are different categories of rules: integrity, production, derivation, reaction and transformations. Integrity rules consist of a constraint assertion. Derivation rules are used to derive conclusion whenever the conditions hold. Production rules produce actions if the conditions hold, while post-conditions must also hold after the execution of actions. A reaction rule is a statement of programming logic that specifies the execution of one or more actions in case of occurrence of a triggering event and satisfaction of its conditions. Optionally, after executing the action, post-conditions may need to be satisfied [22,23]. Rules can be also extracted from existing programs codes that are not expressed in any specific rule dialect by using program slicing techniques [24].

The main problem encountered the implementation of RBS concerns the complete specification of non-redundant and consistent set of rules. This turns out to be a tedious task requiring significant effort of DK engineers. The approaches to verification and validation of RBS do not solve the problem entirely. The verification that is performed after the system designed is both costly and late. Moreover, after introducing the corrections of detected errors, the verification cycle must be repeated. The problem consists in the possibility of introducing new errors through correction of the old ones [17].

The distinctive feature of our system is the design of Rule-based system for data cleaning, which allows one to manage formally rules and their quality.

2.3. Rule Based Data Cleaning

Although DK has been identified as one of the main ingredients for successful DC, the issue of knowledge representation to support DC strategies has not been well dealt with [25]. Entire DQ process will be integrated rule management system in order to achieve clean of data that is as automated, as correct and as quick as possible. Therefore only small number of data would be left for manual inspection and reprocessing [26].

The most current systems based on rules for DC use generated rules to check data in designated tables and mark erroneous records, or to do certain updates of invalid data. They will also store information about the rules violations in order to provide analysis of such data [8, 26].

The incomplete and inconsistent information is a major challenge for RBDC. The quality of a rule base which is an infinity problem that is difficult to solve is an essential issue for the RBDC processing. However, this can't be done effectively using classical logic [27,28].

FOS theory is particularly interesting [29], as it can be a basis towards strong theoretical foundation for developing a rule form that permits us to manage rule and its quality.

2.4. Discussion

The RDBC approaches proposed for both academic research and practical applications have certain persistent limitations related to the following aspects of rule design:

- No practical methodology for RBS is acceptable for RBDC systems because these methodologies are available only for rule production and don't ensuring the quality of rule.
- Lack of formal relation of the rule representation to logic.

Consequently the development of an appropriate RBS for DC is a crucial issue for the final success of RBDC where the rule representation should be of satisfactory expressive power in order to express all of the required rules and be easy to handle and manage rule and its quality. Rule quality control consists in checking for some formal properties that a rule base should exhibit. Let us notice that the most typical formal proprieties are: Consistency, completeness, determinism, redundancy and efficiency [30].

So in this paper, we target at dealing with above limitations by providing an appropriate RBS for DC in which the rule form is based on FOS theory [29].

Our proposal started from the idea that the quality of DC results is determined by the quality of rules used for cleaning. The use of FOS provides a concise and elegant rule representation for the management of rule and its quality.

3. Rule Based Data Cleaning System

An important advantage of the proposed RBDC system is that its design is based on strong rule-based DC metamodel. In addition the metamodel is used as a basis for the development and specification of the uniform rule representation.

3.1. Rule based Data Cleaning Metamodel

The generic meta-model in **Figure 1** which is drawn with the UML notations illustrates the most important information which is worth tracking on RBS for DC. These information are captured through the life cycle of the rule-based DC systems and its supporting data.

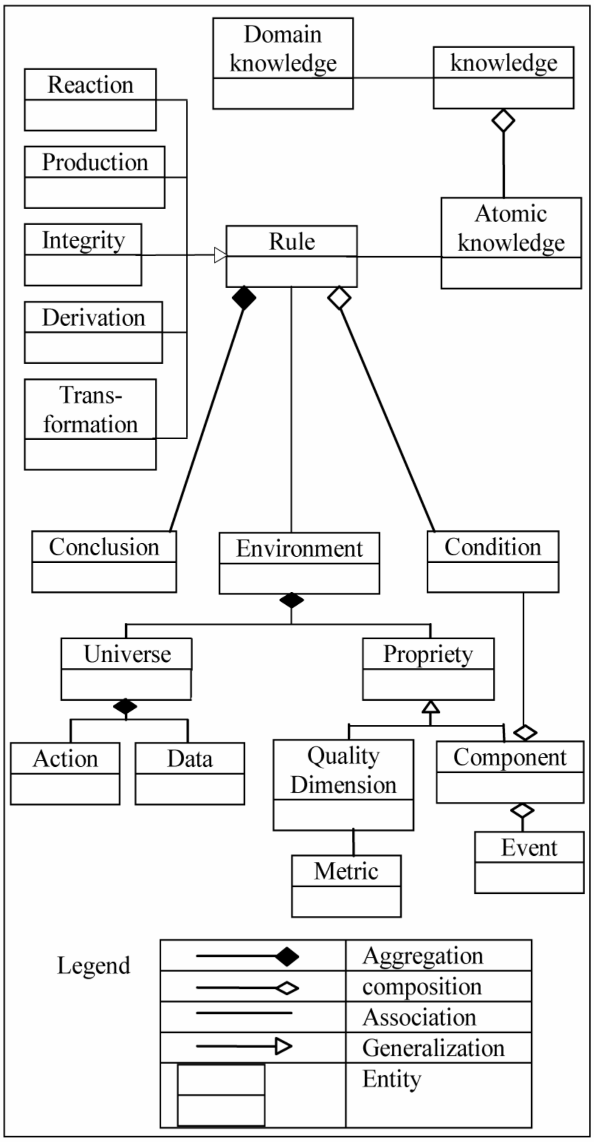


Figure 1. Rule based data cleaning meta-model.

The role of the metamodel is to provide the most components of the system and their relationships. Hence, it offers a good basis to design the RBDC system with the understanding of its functionality and to define a uniform rule representation. The metamodel is divided into two parts, one that presents the knowledge/rule based system and the other that contains the rule management system and rule quality management system. As depicted in **Figure 1**, the main types of rules considered in this metamodel are: reaction, production, integrity, derivation and transformation.

The concept of Component aims at realizing a unified rule form. As the condition and conclusion are the common components of each type of rules, the idea behind the use of Component concept is to represent the uncommon components (for example the component: post-condition is only used for a reaction rule).

3.2. Rule Representation

The expressive power of our rule representation constitutes a crucial feature with respect to the ability of knowledge specification. Hence, it is of primary importance to properly find a uniform rule form that deals with the limitations described above.

Although there are several rule representation built on top of First Order Logic, the concept of Structure that gives a context for determining the value (true, false) of a given formulae has not been taken into account. In this work, we provide rules with the concept of Environment which is an adoption and an adaptation of the Structure in order to create a uniform rule form. Therefore, the rule form is defined as following:

IF Condition **THEN** Conclusion **ON** Environment

The Environment which is built incrementally and iteratively has three main parts: Initial Environment (Initial_Env), Global Environment (Global_Env) and Rule Environment (Rule_Env). It contains three kinds of information: Universe (U), Quality Dimensions (QD) and Component (Cr). The Universe sets a desired target data where the rule will be applied and the actions will be taken if rule is violated. The Quality Dimension is a set of dimensions for the management of the rule quality. The QMS selects suitable metrics from the metrics table for each dimension (see **Figure 2**) and so uses them to evaluate and to verify the quality of rules. The Component is a set of information (triggering event, post-condition and so on) that characterize each type of rule. Thereafter the general form of rule can be rewritten as follows:

IF A **THEN** C **ON** {U, QD, Cr}

where:

- A is a conjunction of atomic conditions. This set is empty in the case of integrity rule.

- C is a set of atomic conclusions.
- U = {DataSet(R), Action} where DataSet(R) is the set of data in which the rule R is applied and the Action is the programs that will be taken when the rule is violated.
- QD is the list of quality dimensions that are used to evaluate the quality of a given rule.
- Cr = {Event, PostCondition…}.

Table 1 shows how the proposed rule representation is wanted to be a general rule form for writing all types of rules in comparison with related works described in [31] that arrange rules types in an hierarchical structure (like Rule Markup Language: RuleML, R2ML) [32].

As the goal of this work is to clean data, it is worth noted that our system defines the Action for each rule differently to the currently RBS where Action is only defined for the reaction and production rules.

3.3. Overview of the RBDC System Components

In this section, we provide an overview of our RBDC system. The proposed system after its installation enables users and knowledge workers to create and to update knowledge and rule through a user interfaces. The rule after its creation is affected to the corresponding temporary rules base, which are used to facilitate the extraction of rule characteristics. These characteristics are exploited by the system to manage the rules and their quality.

Figure 2 shows the design of the proposed RBDC system. The system is a set of knowledge sources, process, tasks, structures and subsystems. It is based on the KBS and RBS technologies. The proposed RBDC system consists of ten elements.

The main knowledge sources used by the RBDC system are:

1) Domain Knowledge: it is the set of knowledge sources from which knowledge can be obtained. In our work, the legacy system, programs and databases referred also to as domain knowledge parts.

2) Knowledge/Rule bases: the Knowledge Base (KB) is temporary area where the knowledge transformations are applied. The Rule Base (BS) is the physical area where rules are stored in order to will be used for DC. Once the knowledge is transformed into a rule, it will be stored in the RB and deleted from the KB.

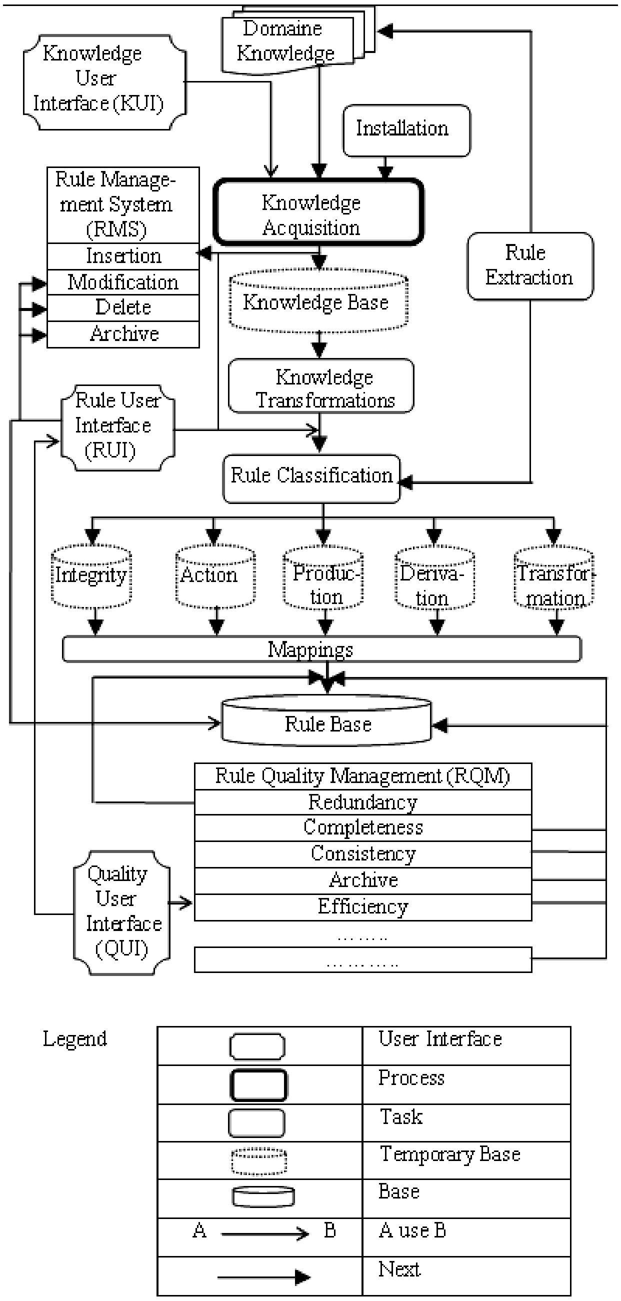


Figure 2. Rule based data cleaning system.

Table 1. Different rule forms.

Type	Condition	Conclusion	Post Condition	Event
Integrity	0	1+	0	0
Reaction	1+	0	0+	1
Production	1+	0	0+	0
Derivation	1+	1	0	0
Transformation	0+	0+	0	0

x = 0,1. x: exactly x x+ = at least

The most typically tasks and processes of the system are:

1) Installation: as the design of RBDC system is independent of the databases to be cleaned; then, it becomes necessary to configure the environment required for performing the RBDC.

2) Knowledge Acquisition: This process focuses on extracting knowledge from experts knowledge, books, documents, and so on by using manual form text or questionnaires.

3) Knowledge Transformation: it provides a set of operations necessary to transform knowledge onto rules.

4) Rule Classification: the goal of this task is to affect each rule to its type (Production, Reaction, Derivation, Integrity and Transformation). For each type, we have defined the corresponding rule template, with which a given rule will be compared.

5) Mapping: it provides a set of transformations rules that permit the expression of each rule of temporary RB with respect to the uniform rule syntax.

6) Rule extraction: it uses the techniques of extraction of rule automatically from computers (Databases, legacy systems, programs).

As the system is interactive, it provides three types of user interfaces. The first type is Knowledge User Interfaces (KUI), which allows us to manage and interact with the informations about knowledge. The second type is Quality User Interface (QUI) which allows managing the quality dimensions and metrics. The third type is Rule User Interface (RUI) which is similar to the KUI.

The RBDC system enables to manage the rule by the Rule Management System (RMS) and its quality by the Quality Management System (QMS). Each system which is considered as a subsystem of the RBDC system is suite of software applications that together make it possible to apply operations on rule and its quality.

More details of the functions of the main elements of the system will be described later in section 5.

As time advances and the sources of DK from which rule is extracted change, rules base must be synchronized with the underlying sources. Thus, after an initial built of our system, RB must be refreshed. For this end, our system allows the refreshment of RB for DC in real time.

4. Description of the System

As we have indicated above, the proposed system provides two subsystems for the management of rules and their quality. These subsystems which are fully integrated with the RBS for DC use the Environment component as basis for rule management.

4.1. Environment

This subsection gives the detail of the construction of the Environment of rule. Let us notice that, contrary to the Global Environment and the Rule one, the Initial Environment is available for any rule.

4.1.1. Initial Environment

The Initial_Env is a set of common quality dimensions which are independent of particular target rule base and applicable and available for each RBS. In particular, it is the set of the most typical proprieties presented in discussion section. It is created during the installation task.

4.1.2. Global Environment

The Global_Env is also a set of quality dimensions specific to the rule base and/or data set to clean. These dimensions can be performing in all rules or a subset (one) of rules. They are introduced to the system through the QUI by the user. They can be also introduced during the insertion or modification of rules through the RUI.

4.1.3. Rule Environment

The Rule_Env is the set of information necessary to the building and execution of rule. These information are introduced by the user through the RUI or extracted from the knowledge base as depicted in **Figure 2**. The rule is considered effective if the Universe (both dataset to clean and action are not empty) is defined.

4.2. Rule Management System

A Rule Management System is a software package that performs and controls the creation, modification, suppression and the archiving of rules. The rules are stored in relational tables. When the rule is not yet applied on data, it can be deleted or archived. The user can be involved to manipulate the rule through the RUI.

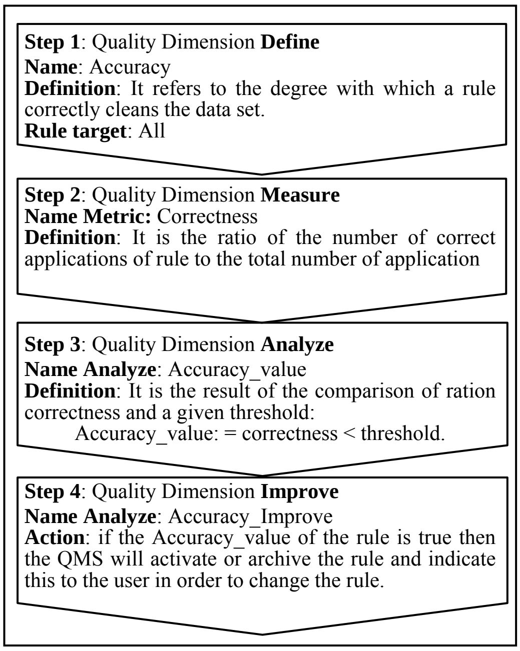
4.3. Quality Management System

As the quality of DC results is determined by the quality of rule, our system integrates a module for rule quality management which can be performed automatically or by the user through the QUI. The QMS follows the continuous life cycle of Total Data Quality Management [33]: Define, Measure, Analyze and Improve rules as essential activities to ensure high quality of data.

Differently to data where the DQ improvement can change the value of data, the QMS can only deactivate or archive rule when it is evaluated poor. Therefore, the user defines the quality dimensions and their quality metrics which allow measuring the quality of rule-through the QUI and after the QMS analyzes the quality of rule in order to decide how to improve it. Let us notice that the use of Environment is to allow an incremental define of quality dimensions and their metrics.

Figure 3 shows how we have defined, measured, analyzed and improved the Accuracy dimension.

Let us notice that the quality dimension can be measured



Step 1: Quality Dimension **Define**
Name: Accuracy
Definition: It refers to the degree with which a rule correctly cleans the data set.
Rule target: All

Step 2: Quality Dimension **Measure**
Name Metric: Correctness
Definition: It is the ratio of the number of correct applications of rule to the total number of application

Step 3: Quality Dimension **Analyze**
Name Analyze: Accuracy_value
Definition: It is the result of the comparison of ration correctness and a given threshold:
Accuracy_value: = correctness < threshold.

Step 4: Quality Dimension **Improve**
Name Analyze: Accuracy_Improve
Action: if the Accuracy_value of the rule is true then the QMS will activate or archive the rule and indicate this to the user in order to change the rule.

Figure 3. Management of accuracy dimension quality.

by combining multiple metrics and also a metric can be used by multiple dimensions. So in our system each metric is implemented as atomic function (code).

5. RBDC Functions

As we have indicated above, this section gives the full details of the processes and tasks described above in subsection 3.3.

5.1. Installation

The installation as we have described above is an important task because it allows to the user to define the information and requirements necessary to the RBDC. Among the most typically information, we cited:

- Location of databases,
- Users authorized to use the rule-based data cl- eaning system,
- List of sources that are parts of the domain kn- owledge,
- A set of ontologically based data quality dimensions and metrics used in the literature and also are important to rules.
- Keywords of the domain knowledge which will be used to regrouping the knowledge.

5.2. Knowledge Acquisition

This process follows two steps: Collection and Storage. The Collection step aims at collecting the information extracted from sources of DK using questionnaires in natural language. The memorization of these information is made during the Storage step according to structured language sentences through the KUI in the KB. Note that the user can store directly these information through the KUI.

5.3. Knowledge Transformations

Differently to the KDS which performs this task during knowledge acquisition process, our system performs it separately (after) to the above process because it needs the intervention of knowledge engineer. The main activities are:

- Creation of knowledge sets according to the keywords knowledge.
- Rewriting of knowledge which does not resp- ect the language sentences.
- Elimination of duplicate knowledge.
- Decomposition of knowledge in atomics know- ledge.
- Elimination of duplicate atomic knowledge.
- Translation of knowledge into rules.

5.4. Rule Extraction

As we have indicated above, the proposed phase allows the extraction of rules from programs, legacy systems and so on by using the data mining and programs slicing techniques. In generally, these techniques extract knowledge as association rules with confidence and support metrics.

5.5. Rule Classification

At this phase, the majority of rules are not in the RB and don't respect the proposed rule form. Then, there is a need to regroup them according to their types. To this end, each rules type has an appropriate rule template with which a given rule will be compared in order to determine its temporary rule base (*i.e.* type). The rule classification simplifies the rule formalization and ensures a higher clarity and consistency of the rule description. This last allows one to identify aspects that are necessary to compute certain quality dimensions of rules. For example the support and confidence metrics are used to compute the accuracy of a rule.

5.6. Rules Mapping

Rules mapping is a task that moves rules from temporary rules bases (integrity, transformation, derivation, production and reaction bases) to rules base (RB), where the rules will be written according to our general rule representtation. The system extracts Condition, Conclusion, Global Environment and Rule Environment from rules before their transformations. The most typically rules transformations are:

1) Transformation 1: Rule_Condition:

It aims at transforming the Condition component of rule in conjunction of atomic conditions.

$$\text{Rule\_Condition (R in Temporary RB)} = \Lambda_{i=0}^{i=n} C_i$$

where Ci is an atomic Condition

This transformation is the same for all types of rules.

2) Transformation 2: Rule_Conclusion

It aims at transforming the Conclusion component of rule in conjunction of atomic conclusions.

$$\text{Rule\_Conclusion(R in Temporary RB)} = \left\{ C_i \middle| C_i \text{ is atomic Conclusion} \right\}$$

This transformation is the same for all the types of rules.

3) Transformation 3: Rule_Univers

The DataSet(R) is extracted directly from the temporary rule but if it is empty, then the system supposed that the rule is applied in the all data. The physical emplacements of Action are also extracted and stored in the universe U. these emplacements can be indicate later by the user through the RUI.

4) Transformation 4: Rule_Components

The Event component of rule, if it exists, should be extracted directly from the temporary rule. The PostCondition component, if it exists, should be mapped by applied Rule_Condition (Transformation 1).

5) Rule 5: Rule_Quality_Dimensions.

It is quite complex and tedious task to completely implement this rule transformation. In order to carry out the transformation of QD, we have designed the Transf_QD algorithm, illustrated in **Figure 4**.

```
Algorithm Transf_QD
Inputs:
        · QUI : Quality User Interface
        · List(QD) : List of QD created during the   installation task
        · QD(R) : list of QD of a given rule R
{QD(R)←∅
        If R is Association Rule then
                {Accuracy←f(support, confidence)}
        {For each QD∈ RB do {
                If QD ∉ List(QD) then
                        {Open QUI
                         Do {
                                Quality Dimension Define of QD
                                Quality Dimension Measure of QD
                                Quality Dimension Analyze of QD
                                Quality Dimension Improve of QD
                                }}
                QD(R)← QD(R) ∪ QD}
        Select QD from list(QD)
        QD(R)← QD(R) ∪ QD}
```

Figure 4. QD Transformation algorithm.

There are two goals underlying this algorithm. Firstly it computes the accuracy value of association rules from the support and confidence metrics. Secondly, it identifies the quality dimensions appropriate to a given rule from its description and added them to the list of QD of the rule. In the case where the QD is not defined, then the algorithm allows to the user to create this dimension and its metrics through a specific QUI. The algorithm also permits the manually selection of quality dimensions that are important from the user point of view.

6. Case Study and Some Experimental Results

In this section, we present some experimental results of the validation of our proposal in the health sector. We study the impact of embedded DK in DC for DQ performances. These experiments show that the proposed RMS and QMS subsystems deal well with some of the limitations of existing rule based data cleaning works.

The most typically components of the architecture of RBDC include:

- Graphical User Interfaces for the dimensions quality, knowledge and rule management.
- RMS and QMS libraries.
- Programs corresponding to the proposed algorithms, process, tasks and transformations.
- Association Rules mining tools.

These components are developed under the JAVA environment. They connect to the Rule-bases, Knowledge bases, Quality tables and Databases to be cleaned via JDBC Bridge. All the data and rules are in the relational schemata. SQL Query is used in our experimentation to perform the extraction and storage of Knowledge/rules. The collection of knowledge from KD is realized through a manual form designed especially for the collection of health information.

The atomic conditions and conclusions of rule in this experimentation are based in the use of attributive logic, which uses attributes for denoting some proprieties (A) of objects (o) and atomic values of attributes (d). It has this form [34]:

$$A(o) \text{ op } d$$

where op is operator.

The results of this case study provide several important observations:

1) Quality of rules: One of the main features of our proposal is the generation of rules and knowledge with their dimensions and metrics quality that allow to the system to discover more interesting and higher quality rules.

2) Expected rules: This kind of rule can be considered as rare, as they would be pruned by many objective interestingness measures (support and confidence). The use

of Universe in rule representation has allowed reducing the number of expected rules.

3) Uncertainty Rule: contrary to the inference engine which does not reasoning with uncertain rules [35], our system reduces the uncertainty by specifying the domain and appropriate dimensions quality of rule by the end user and the incrementally construction of the Global Environment of rule.

4) Rule and Quality Dimensions Groups Validation: this distinguished feature of this work is the ability for the end user to examine groups of rules or of quality dimensions for group/one rules and decide whether to archive or deactivate a group as whole. For example, for a given rule, we can deactivate its appropriate quality dimensions if it is necessary.

5) Autonomous: The QMS and RMS can be performed in on-line and off-line mode. During the initial building of RBDC, the off-line mode is preferable. The clean of data can be also in real time *i.e.* when data is added to the database.

Although the design of the architecture allows one to use the notions of parallelism and pipeline, the time consuming is the major challenge of this proposition especially during the initial building of the RBDC system.

7. Conclusions and Perspectives

This work confirms that conventional software engineering and knowledge engineering are complementary and both essential for developing the increasingly larger systems of today. In this context, this paper describes an approach that advocates the use of knowledge rule-based systems techniques for data cleaning systems. In particular, we aim to make domain knowledge explicit and separate from the system where will be used for data cleaning. For representing domain knowledge, we investigated a uniform and unified rule representation form based on the creation of the environment component that stored information especially for the management of rule and its quality. Besides dealing with some limitations of currently RBS and KBS cited above, the system through the case study allows us to observe many advantages. The major challenge of our proposal is the time consuming. Then, it makes its implementation tedious and complex. This is related to the problem of automation of knowledge acquisition that is not yet resolved.

Finally, once this system is designed, we intend to make it independent from specific database. Therefore, it would be interesting to perform this system with other possible databases. The QMS will also include a feedback loop to enhance the rule quality.

8. References

[1] C. White, “Developing a Universal Approach to Cleansing Customer and Product Data,” *BI Research*, 2008.

[2] E. Friedman-Hill, “Jess in Action, Rule Based Systems in Java,” Meanning Publications CO., 2003.

[3] R. Whiting, “Hamstrung by Defective Data,” *Information Week*, 2006.

[4] A. Vavouras, “A Metadata-Driven Approach for Data Warehouse Refreshment,” Ph.D. Thesis, Der Universität Zürich, Zürich, 2002.

[5] K. Duncan and D. Wells, “Rule Based Data Cleansing for Data Warehousing,” San Diego, 2000.

[6] J. L. Y. Koh, “Correlation-Based Methods for Biological Data Cleaning,” Ph.D. Thesis, Singapore University, 2007.

[7] T. T. P. Thi and M. Helfert, “Discovering Dynamic Integrity Rules with a Rules-Based Tool for Data Quality Analyzing,” *Proceedings of the* 11*th International Conference on Computer Systems and Technologies*, Sofia, 2010, pp. 89-94.

[8] L. Bradji and M. Boufaida, “Knowledge Based Data Cleaning for Data Warehouse Quality,” *Proceeding of ICDIPC* 2011, *Part* 2, *CCIS*, Springer-Verlag Berlin Heidelberg, Vol. 189, 2011, pp. 373-384.

[9] D. Batra and N. A. Wishart, “Comparing a Rule-Based Approach with a Pattern-Based Approach at Different Levels of Complexity of Conceptual Data Modelling Tasks,” *International Journal of Human-Computer Studies*, Vol. 61, No. 4, 2004, pp. 397-419. doi:10.1016/j.ijhcs.2003.12.019

[10] S. Antony and R. Santhanam, “Could the Use of a Knowledge-Based System Lead to Implicit Learning?” *Journal of Decision Support System*, Vol. 43, 2007, pp. 141-151. doi:10.1016/j.dss.2006.08.004

[11] M. Hentea, “Intelligent System for Information Security Management: Architecture and Design Issues,” *Journal of Issues in Informing Science and Information Technology*, Vol. 4, 2007, pp. 29-43.

[12] M. C. Lai, H. C. Huang and W. K. Wang, “Designing a Knowledge-Based System for Benchmarking: A DEA Approach,” *Journal of Knowledge-Based Systems*, Vol. 24, 2011, pp. 662-671. doi:10.1016/j.knosys.2011.02.006

[13] A. Abraham, “Rule-Based Expert Systems,” *Handbook of Measuring System Design*, John Wiley & Sons Ltd, 2005, pp. 909-919.

[14] G. Dondossola, “Formal Methods in the Development of Safety Critical Knowledge-Based Components,” *Proceedings of the KR*’98 *European Workshop on Validation and Verification of KBS*, 1998, pp. 232-237.

[15] A. Onisko, P. Lucas and M. J. Druzdzel, “Comparison of Rule-Based and Bayesian Network Approaches in Medical Diagnostic Systems,” *Proceeding of AIME* 2001, *LNAI*, Springer-Verlag Berlin Heidelberg, 2001, pp. 283-292.

[16] M. S. Abdullah, C. Kimble, I. Benest and R. Paige, “Knowledge-Based Systems: A Re-Evaluation,” *Journal of K-*

nowledge Management, Vol. 10, No. 3, 2006, pp. 127-142. doi:10.1108/13673270610670902

[17] G. J. Nalepa and A. Ligeza, "Prolog-Based Analysis of Tabular Rule-Based Systems with the XTT Approach," *Proceedings of the Nineteenth International Florida Artificial Intelligence Research Society Conference*, 2006, pp. 426-431.

[18] E. Liorens, J. Comas, E. Marti, J. L. Riera, F. Sabater and M. Poch, "Integrating Empirical and Heuristic Knowledge in a KBS to Approach Stream Eutrophication," *Journal of Ecological Modelling,* Vol. 220, 2009, pp. 2162-2172. doi:10.1016/j.ecolmodel.2009.06.012

[19] S. M. M. Soe and M. P. Zaw, "Design and Implementation of Rule-Based Expert System for Fault Management," *Journal of World Academy of Science, Engineering and Technology,* Vol. 48, 2008, pp. 34-39.

[20] C. Angeli, "Diagnostic Expert Systems: From Expert's Knowledge to Real-Time Systems," In: Sajja & Akerkar, Eds., *TMRF e-Book Advanced Knowledge Based Systems*: *Model*, *Applications & Research*, Vol. 1, 2010, pp. 50-73.

[21] G. J. Nalepa and A. Ligeza, "The HEKATE Methodology Hybrid Engineering of Intelligent Systems," *International Journal of Appllied Mathematics and Computer Science*, Vol. 20, No. 1, 2010, pp. 35-53. doi:10.2478/v10006-010-0003-9

[22] A. Paschke and A. Kozlenkov, "A Rule-Based Middleware for Business Process Execution," *Proceedings of Multikonferenz Wirtschaftsinformatik, MKWI* 2008, Munchen, GITO-Verlag, Berlin, 2008, pp. 1409-1420.

[23] M. Ribarić, D. Gašević and M. Milanović, "A Rule-Based Approach to Modeling of Semantically-Enriched Web Services," *Web*4*Web Workshop* 2008, *International workshop on Semantic Web Technologies*, Belgrade, 2008.

[24] F. Tip, "A Survey of Program Slicing Techniques," *Journal of Programming Languages*, Vol. 3, 1995, pp. 121-189.

[25] M. L. Lee, T. W. Ling and W. L. Low, "IntelliClean: A Knowledge-Based Intelligent Data Cleaner," *Proceedings of the sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000, pp. 290-294. doi:10.1145/347090.347154

[26] J. Rodic and M. Baranovic, "Generating Data Quality Rules and Integration into ETL Process," *Proceeding of DOLAP*'09, Hong Kong, 2009, pp. 65-72.

[27] J. Liu, Z, Lu, L. M. López and Y. Xu, " Preference Criterion and Consistency in the Rule-Based System Based on a Lattice-Valued Logic," *Proceeding of Eurofuse Workshop New Trends in Preference Modelling*, 2006, pp. 99-104.

[28] Y. Xu, J. Liu, D. Ruan and T. T. Lee, "On the Consistency of Rule Bases Based on Lattice-Valued First-Order Logic LF(X)," *International Journal of Intelligent Systems*, Vol. 21, No. 4, 2006, pp. 399-424. doi:10.1002/int.20129

[29] S. Hedman,"A First Course in Logic: An Introduction to Model Theory, Proof Theory, Computability and Complexity," Oxford University Press Inc., Edition, 2006.

[30] A. Ligeza and G. J. Nalepa, "A Study of Methodological Issues in Design and Development of Rule-Based Systems: Proposal of a New Approach," *Wiley Interdisciplinary Reviews*: *Data Mining and Knowledge Discovery*, Vol. 1, No. 2, 2011, pp. 117-137. doi:10.1002/widm.11

[31] S. Brockmans, P. Haase, P. Hitzler and R. Studer, "A Metamodel and UML Profile for Rule-Extended OWL DL Ontologies," *Proceedings of* 3*rd European Semantic Web Conference*, *ESWC*, Springer Berlin Heidelberg, Budva, Vol. 4011, 2006, pp. 303-316.

[32] T. F. Gordon, G. Governatori and A. Rotolo, "Rules and Norms: Requirements for Rule Interchange Languages in the Legal Domain," *Lecture Notes in Computer Science*, 2009, Vol. 5858, pp. 282-296. doi:10.1007/978-3-642-04985-9_26

[33] M. P. Angeles and F. García-Ugalde, "A Data Quality Practical Approach," *International Journal on Advances in Software*, Vol. 2 , No. 2 & 3, 2009, pp. 259-273.

[34] A. Ligeza and G. J. Nalepa, "Knowledge Representation with Granular Attributive Logic for XTT-Based Expert Systems," *Proceeding of the Florida AI Research Society Conference-FLAIRS*, 2007, pp. 530-535.

[35] C. C. Chan and Z. Su, "From Data to Knowledge: An Integrated Rule-Based Data Mining System," *Proceedings of the* 17*th International Conference of Software Engineering and Knowledge Engineering*, Taipei, 2005, pp. 508-513.