Ontology-Based System for Conceptual Data Model Evaluation

Zoltan Kazi[1], Ljubica Kazi[1], Biljana Radulovic[1], and Madhusudan Bhatt[2]
[1]Technical Faculty "Mihajlo Pupin" Zrenjanin, University of Novi Sad, Serbia
[2]Department of Computer Science, University of Mumbai, India

Abstract: *Conceptual data modelling is one of the critical phases in Information System (IS) development. In this paper we show the method, software tool and results on automating evaluation of Conceptual Data Model (CDM) from a semantic perspective. The approach is based on mapping ontology with CDM. An ontology that represents domain knowledge and data model are transformed into PROLOG language clauses form and integrated with reasoning rules into the single PROLOG program. The formalization of ontology and the data model is automated within a software tool. Special metrics are defined in aim to enable calculation of semantic marks for data models. Empirical study shows the results of using this tool.*

Keywords: *CDM, evaluation, ontology, formalization, software tool.*

Received February 17, 2014; accepted April 2, 2015; published online October 29, 2015

1. Introduction

Data modelling is an important activity in Information Systems (IS) development, since it includes business domain knowledge representation. This complex activity requires skilled database experts [17]. The results of these activities are data models, theoretically based specifications commonly used for database creation [15]. The data model is a formal abstraction of a real world that is mapped to a database [31]. Various data models are presented and used in the recent few decades, but the most used are Entity Relationship (ER), relational and object-oriented data models.

The ER data model was introduced by Chen [10]. ER models are commonly used in early phases of IS development, usually created as a Conceptual Data Model (CDM) with Computer Aided Software Engineering (CASE) tools. These software systems include tools for conceptual, logical and physical data modelling, integrated with tools to be used in previous phases of IS development (business process modelling and client requirements specification) as well as with tools to be used in phases such as physical data modelling (creating relational data models) and object-oriented modelling (creating class diagrams). Automated transformation of CDM diagrams to a relational data model and class diagrams is supported by CASE tools. Therefore, conceptual data modelling presents a basis for all other commonly used data models and resulting software elements (database and classes). Early activities in IS development, such as: Conceptual data modelling, were recognized as critically important [5], since poor results of these early activities led to failures in final IS product and software [19]. Research in this area is important, because the costs of removing the problem or defect in early phases increase significantly if allocated in later phases of development [8].

Data modelling quality aspects include quality of various data model types, as well as issues regarding the process of data model creation, evaluation and correction [3]. Particularly important activities are related to design and recent ISs research is focused on these issues [34]. The need for reducing problems and failures in the data modelling design require specific metrics and frameworks for measuring the quality of a data model. Application of these metrics and frameworks is to be automated and used within new methodologies and software tools, within IS development [17].

This article presents a method for evaluation of CDM evaluation from semantic aspect, by comparing elements of the CDM with ontology elements. This comparison is performed automatically within PROLOG. Formalization of CDM diagram and ontology is performed by a software tool developed for this purpose. Empirical results of using this method and system are presented.

2. Related Works

Research of evaluating and testing data model correctness, in previous years, was related to ER, relational and object-oriented models. Results of research [32] show that many engineering students have problems with learning how to design a model. To reduce these problems a set of software tools is created to support students' learning model design. Other research is conducted upon CDM comparing with relational models [7], where it was concluded that errors in the CDM modelling process result in normalization problems of relational model. Therefore, it is necessary to minimize possibility of certain types of ER modelling errors, to avoid consequences in relational data model errors. In [1] a formal approach and an automated tool for constructing ontology from the fuzzy relational database is presented. In this study, automated reasoning with fuzzy relational database and ontologies is described.

According to empirical studies [12, 29], the ER data model is still most commonly used in conceptual data modelling. Research [4] presents errors in conceptual modelling as human errors at three levels, i.e., roots of errors: Skill-based, rule-based and knowledge-based. The CODA software prototype was implemented for consulting support to conceptual database design. This software includes heuristics and rules for recognition of typical errors in conceptual data modelling during the process of data model creation. In paper [11] authors reported their experience with of PVS software to formally specify and apply automated reasoning with CDM. The model evaluation metrics were defined in aim to enable comparison of equivalent models in order to direct designer toward a better design [18]. These metrics are quantitative-based, i.e., based on a number of entities, relationships and attributes with certain characteristics [27], complexity of elements and a model [18, 28]. Qualitative-based metrics for quality characteristics could be used for checking completeness, integrity, flexibility, correctness, simplicity, integration implements-ability [24] and preciseness, completeness, consistency, reliability, timeliness, uniqueness, validity [26]. Ontology-based metrics [21] are structure-based and content-based. At last, behavioural-based metrics deal with applicability, maintainability, correctness and performances.

Analysis of recent research results in the evaluation of CDM [25] includes over 50 various proposals to conceptual data modelling evaluation. Less than 20% of the presented methods were empirically validated and none of them is widely accepted in a professional environment. The presented methods mostly focus on error detection and they show lack of measurements metrics and evaluation procedures, agreement of terminology, consistent with standards, guidelines for model improvement, empirical studies of CDM evaluation methods application in practice. According to an analysis [25], only few empirical evaluations of CDM s included action research with collaboration of researchers and practitioners in the field and with practical projects and issues in conceptual data modelling evaluation.

In the field of automating CDM evaluation, various software tools were developed as prototypes. These software tools enable: Analysis of CDM elements quality based on domain ontology [30], comparison of created CDM s with other models [23], automated reasoning on quality of CDM s [11]. Other prototypes refer to the process of CDM creation and improvements by enabling assistance or complete automation in: Consulting support to novice designers for CDM elements quality [4] and automated creation of CDM design [9].

3. CDM Semantic Evaluation

3.1. The Proposed Method

Research [6] suggests some approaches that consider ontologies as a target for reverse engineering for extracting ER, i.e., CDM. Semantics of the relational database can be related with ontology that can be extracted by analyzing the Web pages.

Ontology is often used to capture knowledge about some domain of interest. According to [22] ontology is composed of a finite set of concepts, abstractions that describe the objects of the real world, instances of concepts, i.e., real-world objects, relations between concepts, functions defined over the real-world objects and axioms formalized using first order predicate calculus, needed to determine the meaning of object classes, relations between objects and defined functions over the objects of the real world. Ontology elements and components are presented in several forms [13, 20] such as diagrams or XML notation of ontology languages as Resource Definition Framework (RDF), RDF Schema (RDFS) or Ontology Language (OWL) language. Ontology elements are presented by using other formal languages as first order predicate calculus. Since, ontology is considered that more completely describe business domain knowledge (i.e., semantics) comparing to any data model [15], our approach to CDM evaluation from semantic aspect is based on ontology, i.e., mapping elements of the CDM to ontology elements as shown in Table 1.

Table 1. Mapping ontology elements to CDM (according to [2, 6, 14]).

Ontology	CDM
Class	Entity
Data Property	Attribute
Data Property Range	Attribute Domain
Object Property	Relationship Among Entities
Object Property Range	Relationship Cardinality
Class/Subclass	IS_A Hierarchy

The proposed method is based on creating ontology within an ontology tool, while the CDM is created in a CASE tool. For each problem domain, one ontology must be created and one or more CDM could be compared with the domain ontology. After creation, ontology and data model are both transformed into first order predicate calculus form terms and mapped into PROLOG language clauses form. Reasoning rules for mapping ontology elements to CDM elements and for semantic evaluation were created according to results of research within the EU project [2] and research [14]. Reasoning rules are formally written in PROLOG language form as a set of axioms. A set of axioms formed on reasoning rules is merged with ontology formalization clauses and data model formalization clauses within a software tool. This software tool is developed for models formalization, mapping, integrating and starting PROLOG as a reasoning engine as shown in Figure 1.

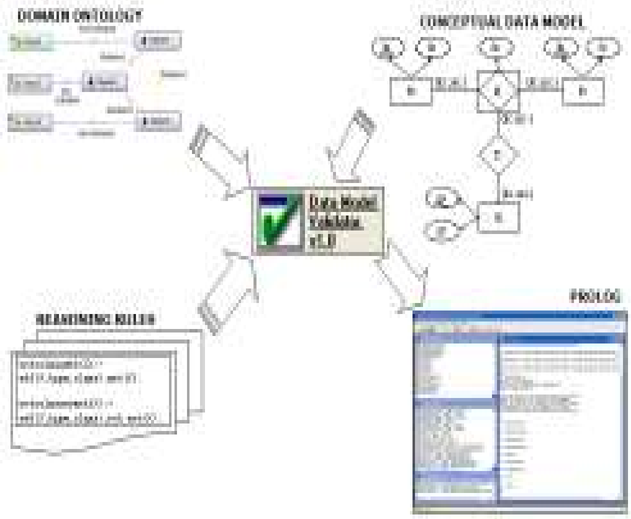

Figure 1. Model of CDM semantic evaluation.

In PROLOG users make queries for each reasoning rule axiom. One PROLOG program consists of ontology clauses, data model clauses and reasoning rules. For each CDM, queries are processed upon reasoning rule axioms and formalized ontology by an automated reasoning PROLOG system. The final activity within the proposed method is computing a semantic evaluation mark for each data model and data model elements. This "semantic evaluation mark" is computed by using our metrics. These metrics are based on metrics from the papers [18, 21, 24, 26, 27, 28].

3.2. CDM Formalization

Data model enables representation of a real world through a set of data entities and their connections [10], that are represented in various ways: Diagram (schema), data dictionary, formal languages representation, such as: Predicate logic calculus [15, 31]. Formal representation of data model schema [31] includes creating sets of meta-model elements. In general, any CDM can be considered as a tuple with entities, relationships, attributes and constraints [5, 16]. Formal presentation of an CDM in our case is presented as a tuple $S=(E, A, R, C, P)$, where E: Is a finite set of entities, A: Is a finite set of attributes, R: Is a finite set of relationships, C: Is a finite set of restrictions concerning attributes domains, relationships constraints, integrity rules for entities, attributes and relationships, and P: Is a finite set of association rules for entities, attributes, relationships and restrictions.

- Formalization of CDM diagram elements is presented within an example. Figure 2 presents CDM diagram example with basic elements.

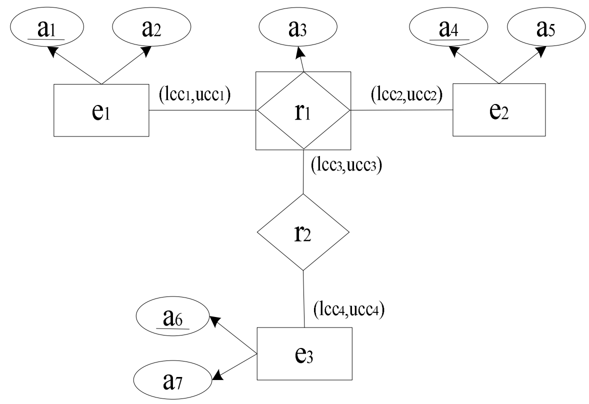

Figure 2. CDM schema.

Set of formalized elements of CDM schema from Figure 2:

- $E=\{e1, e2, e3, r1\}$.
- $A=\{a1, a2, a3, a4, a5, a6, a7\}$.
- $R=\{r1, r2\}$.
- $C=\{id\_atr, mandatory, dom1, dom2, dom3, dom4, dom5, lcc1, ucc1, lcc2, ucc2, lcc3, ucc3, lcc4, ucc4\}$.
- $P=\{p(e1, a1), p(e1, a2), p(r1, a3), p(e2, a4), p(e2, a5), p(e3, a6), p(e3, a7), p(a1, id\_atr), p(a3, id\_atr), p(a6, id\_atr), p(e1, r1), p(e2, r1), p(r1, r2), p(e1, r2), p(r1, lcc1), p(r1, ucc1), p(r1, lcc2), p(r1, ucc2), p(r2, lcc3), p(r2, cc3), p(r2, lcc4), p(r2, ucc4), p(a1, dom1), p(a2, dom2), p(a3, dom3), p(a4, dom4), p(a5, dom5), p(a6, dom6), p(a7, dom7), p(a1, mandatory), p(a2, mandatory), p(a4, mandatory), p(a5, mandatory), p(a6, mandatory), p(a7, mandatory)\}$.

Where id_atr is identifying attribute, *dom*1, *dom*2, *dom*3, *dom*4, *dom*5, *dom*6, *dom*7 are attribute domains, mandatory is a sign for mandatory attributes that are required, *lcc*1, *lcc*2, *lcc*3, *lcc*4 are lower cardinality restrictions, and *ucc*1, *ucc*2, *ucc*3, *ucc*4 are upper cardinality restrictions.

The formalized data model is transformed into PROLOG language clauses form in aim to be mapped with ontology elements in the reasoning rules form. In this transformation, predicate names for elements of S set in PROLOG-like clauses are: "ent" for E set, "atr" for A set, "rel" for R set, "res" for C set and p for P set. This transformation is done by using our software tool. PROLOG clauses for formalized data model elements divided into categories from $S=(E, A, R, C, P)$ are presented in Table 2.

Table 2. Formalized CDM.

CDM Elements	
ent(e1).	p(e2, a4).
ent(e2).	p(e2, a5).
ent(e3).	p(e3, a6).
ent(r1).	p(e3, a7).
rel(r1).	p(a1, idatr).
rel(r2).	p(a3, idatr).
atr(a1).	p(a6, idatr).
atr(a2).	p(e1, r1).
atr(a3).	p(e2, r1).
atr(a4).	p(r1, r2).
atr(a5).	p(e1, r2).
atr(a6).	p(r1, lcc1).
atr(a7).	p(r1, ucc1).
res(idatr).	p(r1, lcc2).
res(mandatory).	p(r1, ucc2).
res(dom1).	p(r2, lcc3).
res(dom2).	p(r2, ucc3).
res(dom3).	p(r2, lcc4).
res(dom4).	p(r2, ucc4).
res(dom5).	p(a1, dom1).
res(dom6).	p(a2, dom2).
res(dom7).	p(a3, dom3).
res(lcc1).	p(a4, dom4).
res(ucc1).	p(a5, dom5).
res(lcc2).	p(a6, dom6).
res(ucc2).	p(a7, dom7).
res(lcc3).	p(a1, mandatory).
res(ucc3).	p(a2, mandatory).
res(lcc4).	p(a4, mandatory),
res(ucc4).	p(a5, mandatory).
p(e1, a1).	p(a6, mandatory).
p(e1, a2).	p(a7, mandatory).
p(r1, a3).	

3.3. Ontology Formalization

The main purpose of an ontology is to capture and enable sharing knowledge about a domain of interest. Ontology is defined as a formal specification of a shared conceptualization of some domain knowledge [13]. Basic characteristics are based on a hierarchy of elements that are instances of concepts established by using different semantic links [22]. Ontology is used to describe words that represent various concepts or as a taxonomy that shows how particular areas of knowledge are related. Basic ontology concepts are: Classes, subclasses, properties, subproperties, domains and ranges [2, 6, 22, 33], i.e., according to [35]: Sets of classes, properties and individuals. Object relations are well defined with object property characteristics. Data properties with data ranges belong to objects that are connected in specific domain. Structure of ontology consists of an collection of OWL/RDF elements [5]. They could be transformed into RDF expression, widely recognized, starting with the World Wide Web Consortium. RDF expression is a collection of triplets: $RDF(S, P, O)$, where S is subject, P is predicate and O is an object. Facts that are described with RDF triplets represent a subject and object relation or even their properties. Figure 3 presents basic domain ontology elements: Classes, objects as class instances and relations of objects. Each object has data property with range that defines the specific data type. These data properties could not be graphically presented at ontology schema, but are presented within an ontology dictionary.

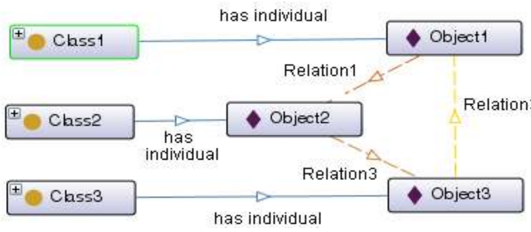

Figure 3. Ontology schema.

The ontology elements as shown in Figure 3 are mapped to predicate logic form, according to [2, 33] and then written in a form of PROLOG-like sentences:

*S= subj:Class*1,
P= rdf:type,
O= owl:Class;
R (*S*, *P*, *O*)→ *R* (*Class*1, *type*, *Class*) →
rdf(*class*1, *type*, *class*).

*S= subj: Relation*1,
P= rdf:type,
O= owl:ObjectProperty;
R (*S*, *P*, *O*)→ *R* (*Relation*1, *type*, *ObjectProperty*)→
rdf(*relation*1, *type*, *objectproperty*).

*S= subj:Object*1,
P= rdf:type,
O= owl:NamedIndividual;
R (*S*, *P*, *O*)→ *R* (*Object*1, *type*, *NamedIndividual*) →
rdf(*object*1, *type*, *namedindividual*).

*S= subj:Object*1,
*P= subj:Relation*1,
*O= subj:Object*2;
R (*S*, *P*, *O*)→ *R* (*Object*1, *Relation*1, *Object*2)→ *rdf*(*object*1, *relation*1, *object*2).

*S= subj:Data*1,
P= rdf:type,
O= owl:DataProperty;
R (*S*, *P*, *O*)→ *R* (*Data*1, *type*, *DataProperty*)→ *rdf*(*data*1, *type*, *dataproperty*).

*S= subj:Data*1,
P= rdf: range,
O= owl:Datatype1;
R (*S*, *P*, *O*)→ *(Data1, range, Datatype1) → rdf(data1, range, datatype1).*

S= subj:Object1,
P= rdf: datapropertyassertion,
O= owl:Data1;
R (*S*, *P*, *O*)→ *R* (*Object*1, *datapropertyassertion*, *Data*1)→
rdf(*object*1, *datapropertyassertion*, *data*1).

This mapping of ontology elements in a form of triplets is shown for characteristic types of XML elements that are present in RDF formats. The complete set of ontology RDF triplets, which are mapped into predicate logic terms and then transformed to PROLOG language sentences, is listed below:

- *rdf*(*class*1, *type*, *class*).
- *rdf*(*class*2, *type*, *class*).
- *rdf*(*class*3, *type*, *class*).
- *rdf*(*object*1, *type*, *namedindividual*).
- *rdf*(*object*2, *type*, *namedindividual*).
- *rdf*(*object*3, *type*, *namedindividual*).

- *rdf*(*relation*1, *type*, *objectproperty*).
- *rdf*(*relation*2, *type*, *objectproperty*).
- *rdf*(*relation*3, *type*, *objectproperty*).
- *rdf*(*object*1, *relation*1, *object*2).
- *rdf*(*object*1, *relation*2, *object*3).
- *rdf*(*object*3, *relation*3, *object*1).
- *rdf*(*data*1, *type*, *dataproperty*).
- *rdf*(*data*2, *type*, *dataproperty*).
- *rdf*(*data*3, *type*, *dataproperty*).
- *rdf*(*data*1, *range*, *datatype*1).
- *rdf*(*data*2, *range*, *datatype*2).
- *rdf*(*data*3, *range*, *datatype*3).
- *rdf*(*object*1, *datapropertyassertion*, *data*1).
- *rdf*(*object*2, *datapropertyassertion*, *data*2).
- *rdf*(*object*3, *datapropertyassertion*, *data*3).

3.4. Reasoning Rules

Reasoning rules for semantic evaluation of the CDM are formed according to [2, 6, 14], where the authors have defined a mapping for ontology elements to CDM as shown in Table 1. Classes from ontology are mapped to entities in the data model, object data properties to data model attributes, data property ranges to attribute domains, ontology object properties to ERs, object properties to the cardinality of relationships, taxonomy of classes and subclasses refers to an IS_A hierarchy in the data model. According to [2, 6, 14] in this paper we present predicate logic notation of reasoning rules that are merged within our software tool with ontology and data model formalization in an aim to be processed in PROLOG. These reasoning rules enable performing queries in PROLOG that compute answers in the form of sets of data model or ontology elements that have certain characteristics.

- *Rule* 1: Ontology classes that are covered by entities in CDM. For each class from ontology must be definied named entity set in CDM:

$$ontoclassent(X):\text{-}rdf(X, type, class), ent(X) \tag{1}$$

- *Rule* 2: Ontology classes that are not covered by entities in CDM:

$$ontoclassnoent(X):\text{-}rdf(X, type, class), not\ ent(X) \tag{2}$$

- *Rule* 3: Data properties from ontology that are covered by attributes in CDM. For each data property in ontology must be defined named attribute in data model:

$$ontodataatrib(X):\text{-}rdf(X, type, dataproperty), atr(X) \tag{3}$$

- *Rule* 4: Ontology data properties that are not covered by attributes in CDM:

$$ontodatanoatrib(X):\text{-}rdf(X, type, dataproperty), not\ atr(X) \tag{4}$$

- *Rule* 5: Data property ranges from ontology that are covered with attributes in data model that have defined data types in CDM:

$$\begin{aligned}&ontodataatribtype(X, Y):\text{-}rdf(X, type, dataproperty),\\&rdf(X, range, Y), atr(X), res(Z), p(X, Z), datatype(Y, Z)\end{aligned} \tag{5}$$

- *Rule* 6: Ontology object property ranges for classes that match with relationship cardinality restrictions in CDM for those entities that are covered with ontology classes:

$$\begin{aligned}&ontocard(C, OP, CD1, CD2):\text{-}rdf(OP, type, objectproperty),\\&rdf(C, OP, CD1), rdf(C, OP, CD2), ent(E1), ent(E2), rel(R),\\&p(E1, R), p(R, E2), (p(CD1, R), p(CD2, R)), (p(R, CD1),\\&p(R, CD2)), (E1=C, E2=C), R=OP, not\ CD1=CD2\end{aligned} \tag{6}$$

- *Rule* 7: Object properties from ontology that are covered by relationships in CDM. For each object property from ontology must be declared named relationship in CDM:

$$ontoobjproprel(X):\text{-}rdf(X, type, objectproperty), rel(X) \tag{7}$$

- *Rule* 8: Object properties from ontology that are not covered by relationships in CDM:

$$ontoobjpropnorel(X):\text{-}rdf(X, type, objectproperty), not\ rel(X) \tag{8}$$

- *Rule* 9: Appropriate entity-relationship in CDM exists for every object property of a class instance from ontology:

$$\begin{aligned}&ontoobjpropger(C1, OP, C2, E1, ER, E2):\text{-}\\&rdf(OP, type, objectproperty), rdf(C1, type, class),\\&rdf(X1, classassertion, C1), rdf(X1, type, namedindividual),\\&rdf(C2, type, class), rdf(X2, classassertion, C2),\\&rdf(X2, type, namedindividual), rdf(X1, OP, X2),\\&ent(E1), ent(ER), ent(E2), rel(R1), rel(R2),\\&p(E1, R1), p(R1, ER), p(ER, R2), p(R2, E2), OP=ER\end{aligned} \tag{9}$$

- *Rule* 10: Ontology classes and subclasses that are covered by IS_A hierarchy entities in CDM. For any ontology class must be defined named entity super-class in data model, and each ontology subclass is presented with entity subtype, with restriction that subtypes in data model must be different objects:

$$\begin{aligned}&ontosubclassisa(X, X1, X2):\text{-}rdf(X, type, class),\\&rdf(X1, subclass, X), rdf(X2, subclass, X), ent(X),\\&ent(X1), ent(X2), p(X,Y), p(Y,X1), p(Y, X2),\\&not\ X1=X2\end{aligned} \tag{10}$$

- *Rule* 11: Ontology classes and subclasses that are not covered by IS_A hierarchy entities in CDM:

$$\begin{aligned}&ontosubclassnoisa(X, X1, X2):\text{-}rdf(X, type, class),\\&rdf(X1, subclass, X), rdf(X2, subclass, X),\\&not\ X1=X2, not\ ontosubclassisa(X, X1, X2)\end{aligned} \tag{11}$$

Where *X*, *X*1, *X*2, *Y*, *E*1, *E*2, *C*1, *C*2, *CD*1, *CD*2, *R*1, *R*2, *OP* and *ER* represents variables. *type*, *class*, *subclass*, *objectproperty*, *dataproperty*, *range*, *namedindividual*, class assertion are constant values. *rdf*, *ent*, *atr*, *rel*, *res*, *p*, *ontoclassent*, *ontoclassnoent*, *ontodataatrib*, *ontodatanoatrib*, *ontodataatribtype*, *ontocard*, *ontoobjproprel*, *ontoobjpropger*, *ontosublassisa* and *ontosublassnoisa* are symbols for predicates.

3.5. Metrics for CDM Semantic Evaluation

As previously mentioned in the method description (3.1), the final activity in the CDM evaluation is computing semantic mark of the model based on metrics. These are qualitative-based metrics that include content-based characteristics such as completeness, integrity and correctness (similar to those presented in [21, 24, 26]). Final rank for each CDM represents an average value of four marks for ontology and data model element groups (each called "ontology mark"): Classes and entities, relationships and object properties, attributes and data properties, classes with subclasses and IS_A hierarchy. For each model element in metric (according to research [21]), a "weight factor" K_T is given, where T represents a conceptual data element type. This "weight factor" represents a quantitative expressed significance of an element in the analysis of the whole CDM.

$$OM = \frac{K_E \cdot OM_E + K_A \cdot OM_A + K_R \cdot OM_R + K_{SC} \cdot OM_{SC}}{4} \tag{12}$$

Explanation of equation elements are OM ontology mark for each data model, OM_E ontology mark for entities, OM_A ontology mark for attributes, OM_R ontology mark for relationships and OM_{SC} ontology mark for super-class entities and sub-class entities. K_E, K_A, K_R and K_{SC} weight factors. Note, in this paper within an empirical study all weight factors have values=1, but according to evaluator criteria, they could be different and could take any decimal value from interval [0, 1].

Ontology mark for entities is calculated as:

$$OM_E = \frac{\sum_{n_E}^{i_E=1} E(Rule\,1)_{i_E} \cdot 100}{\sum_{n_E}^{i_E=1} E(Rule\,1)_{i_E} + \sum_{n_E}^{i_E=1} E(Rule\,2)_{i_E}} \tag{13}$$

Ontology mark for attributes is calculated as:

$$OM_A = \frac{\dfrac{\sum_{n_A}^{i_A=1} A(Rule\,3)_{i_A} + \sum_{n_A}^{i_A=1} A(Rule\,5)_{i_A}}{2} \cdot 100}{\sum_{n_A}^{i_A=1} A(Rule\,3)_{i_A} + \sum_{n_A}^{i_A=1} A(Rule\,4)_{i_A}} \tag{14}$$

Ontology mark for relationships is calculated as:

$$OM_R = \frac{\dfrac{\sum_{n_R}^{i_R=1} R(Rule\,6)_{i_R} + \sum_{n_R}^{i_R=1} R(Rule\,7)_{i_R} + \sum_{n_R}^{i_R=1} R(Rule\,9)_{i_R}}{3} \cdot 100}{\sum_{n_R}^{i_R=1} R(Rule\,7)_{i_R} + \sum_{n_R}^{i_R=1} R(Rule\,8)_{i_R}} \tag{15}$$

Ontology mark for classes and sub-classes is calculated as:

$$OM_{SC} = \frac{\sum_{n_{SC}}^{i_{SC}=1} SC(Rule\,10)_{i_{SC}} \cdot 100}{\sum_{n_{SC}}^{i_{SC}=1} SC(Rule\,10)_{i_{SC}} + \sum_{n_{SC}}^{i_{SC}=1} SC(Rule\,11)_{i_{SC}}} \tag{16}$$

Minimum values for OM, OM_E, OM_A, OM_R and OM_{SC} marks are zero, maximum values are 100, both for particular mark and for the total semantic mark of a complete CDM.

4. Software Tools

In the process of CDM design evaluation various software tools are used. We propose an integration of particular software tools as well as using our integration software tool that simplifies and speeds up the whole process. The basis of integration is using XML as a form of results that some software tools produce.

We propose empirically tested and used tools as part of the proposed system:

- Protégé as an ontology tool that enables saving the result of designing in OWL language or RDF output form with structure that is actually an XML document. This software tool was developed by Stanford University.
- Power Designer tool is a CASE tool that includes conceptual data modelling tool for CDM creating. This software enables saving results of modelling as *.CDM file, which has an XML specific structure. This software tool was developed by SYBASE.
- Data model validatoras shown in Figure 4 our integration and transformation software tool especially developed for ontology and data model formalization, mapping and integrating, to create a PROLOG program to be an input in starting PROLOG.

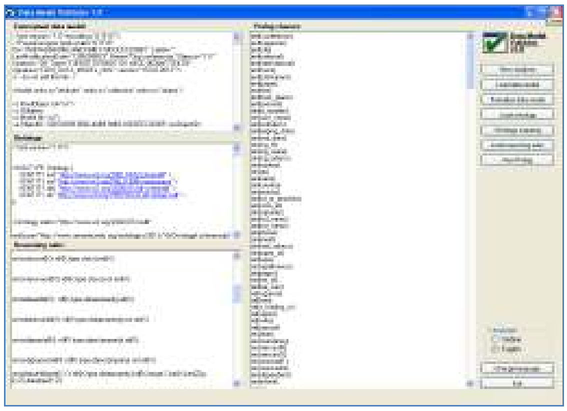

Figure 4. Data model validator software tool.

- PROLOG automated reasoning tool, for executing a program that is actually merged domain ontology formalization, CDM formalization and our reasoning rules.

The proposed system of tools is used within the proposed method by following certain steps of method application:

1. Creating ontology in Protégé editor for a domain of interest.
2. Creating CDM in power designer tool.
3. Running the data model validator for loading ontology, CDM and reasoning rules.
4. Mapping ontology and CDM formalization within data model validator into PROLOG facts that have to be merged with reasoning rules in a single program.
5. Running PROLOG listener for consulting a program that consists ontology, data model and reasoning rules.
6. Making goals in PROLOG to activate automated reasoning and get results with output unification.
7. Entering the PROLOG goals results into metric that calculates a semantic mark for the CDM.

5. Empirical Study

Empirical study of the proposed model is conducted as a laboratory experiment with undergraduate students' data models collected from the practical exam. Participants of this research are students from the University of Novi Sad, Technical faculty "Mihajlo Pupin" in Zrenjanin, Serbia, studying this course at the third semester of undergraduate studies of information technology engineering.

Empirical study is based on analysis and evaluation of 165 students' works (CDM models) created at the same exam content. The task in the exam was to create an CDM for organizing international conferences. We created a single ontology to present the specified domain knowledge. Mapping ontology for organizing international conferences into PROLOG clauses with data model validator tool resulted in more than 330 facts that represent ontology RDF triplets. Each student's data model was loaded in data model validator tool, formalized and integrated with ontology triplets and reasoning rules. PROLOG clauses for data models were different, from minimally 160, to more than 240 program sentences. Integrated PROLOG programs have, from 500 clauses for the smallest data model to more than 600 clauses for the largest data models. All these programs were loaded into PROLOG for individual executing and making results upon queries.

Statistics for testing of each CDM element ontological correctness are presented in Table 3. Ontology classes are mapped to entities in CDM by 92,68%, ontology data properties with attributes by 41,43%, data property ranges are mapping 58,57% attributes data types, ontology object properties with 64,75% relationships in the data model, relationship cardinality mapping with ontology object property ranges is 32,69%. Ontology classes are mapped with super-class entities in the data model with 57,00% and ontology sub-classes are covered by subclass entities with 55,50%.

Table 3. Result of testing CDM elements.

Elements from Ontology and Data Model	Average Number of Elements Per Model	Total Number of Elements In Ontology	Ontology Elements Coverage (%)
Entities Mapped to Ontology Classes	8, 34	9	92, 68
Attributes Mapped to Ontology Dataproperties	12, 43	30	41, 43
Attributes Data Types Mapped to Data Property Range	17, 57	30	58, 57
Relationships Mapped to Ontology Object Classes	5, 18	8	64, 75
Relationship Cardinality Mapped to Ontology Object Property Ranges	5, 23	16	32, 69
Superclasses of IS_A Hierarchy Mapped to Ontology Classes	0, 57	1	57, 00
Subclasses of IS_A Hierarchy Mapped to Ontology Subclasses	1, 11	2	55, 50

Data presented in Table 3 shows the result of the semantic correctness analysis for 165 students' data models. Overall ontology mark for each CDM as a whole was computed as an average value of each category mark.

Analysis of these results shows that the best ranked data model has 89,7% correctness of checked semantic characteristics and the worst model was at 36,1%. Mapping average marks for entities was in the range from 66,7% to 100%, for attributes and their domains it was from 26,7% to 67,8%. Relationship mapping is in the range of 15,6% to 81,3%, and IS_A hierarchy is mapped in the range from 0% to 100%. An average mark for all CDMs is 61,73% of semantic correctness.

6. Conclusions and Future Works

Many research efforts in the area of CDM evaluation were focused on creating new methodologies and frameworks. Still, there is not much agreement on creating a single framework or a standard methodology for data model evaluation, particularly for CDMs. Proposed approaches are still in the domain of theory and only 20% of them are empirically evaluated. Recent years emphasize focus on automation of data model evaluation.

This paper shows the results of research that integrate using CASE tool for CDM creation and ontology editor for ontology domain modelling with a set of rules for semantic evaluation of the CDM model. We have developed a software tool named data model validator for CDM formalization and ontology mapping and integration with reasoning rules. PROLOG is used in automated reasoning and creating answers for queries related to application of reasoning rules. Results of automated reasoning in PROLOG are used for calculating the semantic mark of data model elements.

Empirical study for testing the implementation of method and developed tool results in statistical data. Usability of developed software tool in higher education area is shown. This approach is applicable in situations where an ontology is created as a basis for evaluation of a group of conceptual models with the same semantics. Weight factors in metric equations enable each element category significance adjustment and influence to final semantic evaluation mark for each CDM.

The contributions of this research include a method for evaluating the semantic aspect of CDM. This method is based on ontology mapping, a procedure for formalization of the CDM and ontology, reasoning rules for mapping CDMs with ontology, development of specific transformation and integration tool for semantic validation of CDM, metrics for evaluation of semantic aspects of quality of CDM. The developed software tool is scalable and flexible, since it was implemented by separating reasoning rules from reasoning logic. Metrics equations are adjustable to evaluator's marking criteria.

Limitations of this research are related to the developed software tool. The tool has an internal application logic for parsing CDM elements from XML created in the particular CASE tool. Other types of CASE tools i.e. their output formats for CDMs are not supported, particularly those data models that are not presented by XML format. Another limitation is related to the set of reasoning rules, that is focused on ontology mapping with the CDM. The limitation of this research is also related to an empirical study. Verification of the proposed approach and developed tool is performed upon a limited number of undergraduate students' works.

Future work includes modification of our software tool to other types of data model file formats, extension of reasoning rules to enable both syntax and semantic verification, in aim to enable more complete data model verification, increasing the automation level of CDM evaluation, development of the consultation expert module for more user friendly presentation of data modelling errors and suggestions for improvements. Need for verification of proposed approach and software tool with IT professionals that have more experience on database design is one of future research directions.

References

[1] Almeida M. and Barbosa R., "Ontologies in Knowledge Management Support: A Case Study," *Journal of the American Society for Information Science and Technology*, vol. 60, no. 10, pp. 2032-2047, 2009.

[2] Antoniou G., Damasio C.V., Grosof B., Horrocks I., Kifer M., Maluszynski J., and Patel-Schneider P.F., "Combining Rules and Ontologies," available at: http://rewerse.net/deliverables/m12/i3-d3.pdf, last visited 2005.

[3] Batini C. and Scannapieco M., *Data Quality*, Springer, Heidelberg, 2006.

[4] Batra D. and Antony S., "Consulting Support During Conceptual Database Design in the Presence of Redundancy in Requirements Specifications: An Empirical Study," *International Journal of Human-Computer Studies*, vol. 54, no. 1, pp. 25-51, 2011.

[5] Benslimane S., Malki M., and Bouchiha D., "Deriving Conceptual Schema from Domain Ontology: A Web Application Reverse Engineering Approach," *The International Arab Journal of Information Technology*, vol. 7, no. 2, pp. 167-176., 2010.

[6] Benslimane S., Malki M., Rahmouni M., and Rahmoun A., "Towards Ontology Extraction from Data-Intensive Web Sites: An HTML Forms-Based Reverse Engineering Approach," *The International Arab Journal of Information Technology*, vol. 5, no. 1, pp. 34-44, 2008.

[7] Bock D., "Entity-Relationship Modelling and Normalization Errors," *Journal of Database Management*, vol. 8, no. 1, pp. 4-12, 1997.

[8] Boehm B., *Software Engineering Economics*, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1981.

[9] Brdjanin D. and Maric S., "An Approach to Automated Conceptual Database Design Based on the UML Activity Diagram," *Computer Science and Information Systems ComSIS*, vol. 9, no. 1, pp. 250-283, 2012.

[10] Chen P., "The Entity-Relationship Data Model – Toward a Unified View of Data," *ACM Transactions on Database Systems*, vol. 1, no. 1, pp. 9-36, 1976.

[11] Choppella V., Sengupta A., Robertson E., and Johnson S., "Preliminary Explorations in Specifying and Validating Entity-Relationship Models in PVS," *in Proceedings of the 2nd Workshop on Automated Formal Methods*, Atlanta, USA, pp. 1-10, 2007.

[12] De Lucia A., Gravino C., Oliveto R., and Tortora G., "An Experimental Comparison of ER and UML Class Diagrams for Data Modelling," *Journal of Empirical Software Engineering*, vol. 15, no. 5, pp. 455-492, 2009.

[13] Dekdouk A., "Ontology-Based Intelligent Mobile Search Oriented to Global e-Commerce," *The International Arab Journal of Information Technology*, vol. 7, no. 1, pp. 28-33, 2010.

[14] El-Ghalayini H., Odeh M., McClatchey R., and Solomonides T., "Reverse Engineering Ontology to Conceptual Data Models," available at: https://arxiv.org/ftp/cs/papers/0412/0412036.pdf, last visited 2005.

[15] Elmasri R. and Navathe S., *Fundamentals of Database Systems*, Addison Wesley, 2007.

[16] Emer M.C., Vergilio S., and Jino M., "Testing Relational Database Schemas with Alternative Instance Analysis," available at: http://ecommons.luc.edu/cgi/viewcontent.cgi?article=1036&context=cs_facpubs#page=386, last visited 2008.

[17] Formica A. and Missikoff M., "Correctness of IS-A Hierarchies in Object-Oriented Database Schemas," *in Proceedings of the 4th International Conference on Extending Database Technology Cambridge*, UK, pp. 231-244, 2006.

[18] Gray R., Carey B., McGlynn N., and Pengelly A., "Design Metrics for Database Systems," *BT Technology Journal*, vol. 9, no. 4, pp. 69-79, 1991.

[19] Hannola L. and Ovaska P., "Challenging Front-End-of-Innovation in Information Systems," *Journal of Computer Information Systems*, vol. 52, no. 1, pp. 66-75, 2011.

[20] Horridge M., *A Practical Guide to Building OWL Ontologies Using Protege 4 and CO-ODE Tools Edition 1.2*, University of Manchester, 2009.

[21] Kesh S., "Evaluating the Quality of Entity Relationship Models," *Information and Software Technology*, vol. 37, no. 12, pp. 681-689, 1995.

[22] Lammari N. and Elisabeth M., "Building and Maintaining Ontologies-A Set of Algorithms," *Journal of Data and Knowledge Engineering*, vol. 48, no. 2, pp. 155-176, 2004.

[23] Moody D. and Shanks G., "Improving the Quality of Data Models: Empirical Validation of a Quality Management Framework," *Information Systems*, vol. 28, no. 6, pp. 619-650, 2003.

[24] Moody D., "Metrics for Evaluating the Quality of Entity Relationship Models," *in Proceedings of the 17th International Conference on Conceptual Modelling ER*, Singapore, pp. 211-225, 1998.

[25] Moody D., "Theoretical and Practical Issues in Evaluating the Quality of Conceptual Models: Current State and Future Directions," *Data and Knowledge Engineering*, vol. 55, no. 3, pp. 243-276, 2005.

[26] Piprani B. and Ernst D., "A Model for Data Quality Assessment," *in Proceedings of the OTM Workshops, Springer Berlin/Heidelberg*, pp. 750-759, 2008.

[27] Piattini M., Genero M., and Calero C., "Data Model Metrics," *in Proccedings of UK Academy of Information Systems Annual Conference*, Cardiff, pp. 1-18, 2000.

[28] Piattini M., Genero M., Calero C., Polo M., and Ruiz F.,"*: Advanced Database Technology and Design*, Eds. Mario Piattini & Oscar Díaz, Artech House, 2000.

[29] Shoval P., "Experimental Comparisons of Entity-Relationship and Object-Oriented Data Models," *Australasian Journal of Information Systems*, vol. 4, no. 2, pp. 74-81, 1997.

[30] Sugumaran V. and Storey V., "The Role of Domain Ontologies in Database Design: An Ontology Management and Conceptual Design Environment," *ACM Transactions on Database Systems*, vol. 31, no. 3, pp. 1-28, 2006.

[31] Ullman J., Garcia–Molina H., and Widom J., *Database Systems-The Complete Book*, Department of Computer Science, Stanford University, Prentice Hall, New Jersey, 2002.

[32] Van der Schaaf H., Tramper J., Vermue M., and Hartog R., "Support of Modelling in Process-Engineering Education," *Computer Applications in Engineering Education*, vol. 14, no. 3, pp. 161-168, 2006.

[33] Vysniauskas E. and Nemuraite L., "Transforming Ontology Representation from OWL to Relational Database," *Information Technology and Control*, vol. 35, no. 3A, pp. 333-343, 2006.

[34] Wang S. and Wang H., "Towards Innovative Design Research in Information Systems," *Journal of Computer Information Systems*, vol. 51, no. 1, pp. 11-18, 2010.

[35] Xue X., Yuping Y., and Hao W., "Optimizing Ontology Alignments by using NSGA-II," *The International Arab Journal of Information Technology*, vol. 12, no. 2, pp. 176-182, 2015.

Zoltan Kazi is a lecturer at University of Novi Sad, at Technical faculty "Mihajlo Pupin" Zrenjanin, Serbia. He received his PhD degree in computer science from the Novi Sad University in 2014. His research interests include knowledge based systems, automated reasoning systems, ontology engineering, business intelligence, complex databases, information system modeling and development.

Ljubica Kazi is a lecturer at University of Novi Sad, at Technical faculty "Mihajlo Pupin" Zrenjanin, Serbia. She received MSc degree in computer science from the Novi Sad University in 2009. She is a PhD student in computer science at Novi Sad University, Serbia. Her research interests include information system modeling, business intelligence and distributed information systems development.

Biljana Radulovic is a full professor at University of Novi Sad, at Technical faculty "Mihajlo Pupin" Zrenjanin, Serbia. She received PhD degree in technical sciences and informatics from Novi Sad University in 1998. Her research interests include information systems, data warehouse, business intelligence, complex databases and distributed systems development.

Madhusudan Bhatt is a senior professor of Department of Computer Science at R.D. National College affiliated to University of Mumbai, India. He received the PhD degree in computer science from Mumbai University in 1989. He was associated with K.C.College, University of Mumbai, India, for more than 30 years as the head of Department of Computer Science. His research interests include system analysis and design, business process modeling and implementation of information systems.