American Journal of Networks and Communications
2015; 4(3-1): 16-21
Published online January 16, 2015 (http://www.sciencepublishinggroup.com/j/ajnc)
doi: 10.11648/j.ajnc.s.2015040301.14
ISSN: 2326-893X (Print); ISSN: 2326-8964 (Online)

Ontology based fuzzy query execution

Geetanjali tyagi, kumar kaushik, Arnika Jain, Manish Bhardwaj

Department of Computer science and Engineering, SRM University, NCR Campus, Modinagar, Ghaziabad, India

Email address:
git101288@gmail.com (G. Tyagi), kumarkaushik26@gmail.com (K. Kaushik) , jain.arnika2009@gmail.com (A. Jain), aapkaapna13@gmail.com (M. Bhardwaj)

To cite this article:
Geetanjali tyagi, kumar kaushik, Arnika Jain, Manish Bhardwaj. Ontology Based Fuzzy Query Execution. *American Journal of Networks and Communications*. Special Issue: Ad Hoc Networks. Vol. 4, No. 3-1, 2015, pp. 16-21. doi: 10.11648/j.ajnc.s.2015040301.14

Abstract: Database engineering has been progressed up to the Relational database stage. Fuzzy information administration in databases is a complex process in view of adaptable information nature and heterogeneous database frameworks. Relational Database Management System (RDBMS) can just handle fresh information but cannot handle precise data information. Structured Query Language (SQL) is a very powerful tool but can handle data which is crisp and precise in nature. It is not able to fulfill the requirements for information which is indeterminate, uncertain, inapplicable and imprecise and vague in nature. The goal of this work is to use Fuzzy technique in RDBMS. But, Fuzzy Relational Database Management System (FRDB) requires complex data structures, in most cases, are dependent on the platform in which they are implemented. A solution that involves representing an FRDB using an Ontology as an interface has been defined to overcome this problem. A new Fuzzy Query Ontology is proposed in this dissertation with implementation. The implementation layer, which is responsible for parsing and translating user requests into the corresponding DB implementations in transparent, is required to establish communication between the Ontology and the relational databases management system (RDBMS). This ontology defines a framework for storing fuzzy data by defining those using classes, slots, and instances. An Ontology is an explicit and formal specification of a conceptualization. Ontologies provides a shared understanding of a domain which allows interoperability between semantics.

Keywords: Fuzzy, Ontology, Fuzzy Data, Metadata, DBMS Catalog, Fuzzy Knowledge Representation, Ontology, Databases, Database Modeling

1. Introduction

Relational Database Management System (RDBMS) is used to store crisp data and SQL Language is used to intact with database. A Relational Database Management System (RDBMS) extension for representing fuzzy data is obviously not a new problem. Relational Database can store only crisp, precise data, but can able to store fuzzy values. Complexity normally arises from uncertainty in the form of ambiguity. The computerized system is not capable of addressing uncertain, imprecise data and ambiguous issues whereas, the human have the capacity to reason “approximately”. As a result, human, when interacting with the database, want to make complex queries that have a lot of vagueness present in it. So, database which is in use cannot store uncertain data. But in real situations, these are not crisp and deterministic and therefore, cannot be described precisely. The conventional database management system does not handle imprecise, incomplete or vague information such as very high, approximately some values. To triumph over this problem, the fuzzy database system has been introduced. And SQL Language works with conventional database, so in order to interact with fuzzy database we need to extend SQL into language. This paper is organized as follows: Fuzzy Relational Database is presented in section 2. Section 3 The GEFRED Model. Section 4 ONTOLOGY. Section 5 the implementation detail of criteria that follow.

2. Fuzzy Relational Database

2.1. Fuzzy Relational Database Model

The basic model of fuzzy relational databases is considered the simplest one, and it consists of adding a grade, normally in the [0, 1] interval, to each instance (or tuple). This makes keeping database data homogeneity possible. Nevertheless, the semantic assigned to this grade will determine its usefulness, and this meaning will be utilized in the query processes. This grade may have the meaning of membership degree of each tuple to the relation (Giardina,

1979; Mouaddib, 1994). But it may mean something different, such as the dependence strength level between two attributes, thus representing the relation between them (Baldwin, 1983), the fulfillment Prade, 1997) of each tuple in the relation, among others.The main problem with these fuzzy models is that they do not allow for the representation of imprecise information about a certain attribute of a specific entity (such as the "tall" or "short" values for a height attribute). Besides, the fuzzy character is assigned globally to each instance (tuple), making it impossible to determine the specific fuzzy contribution from each constituting attribute.

2.2. Fuzzy Logic

Fuzzy Logic is a form of many valued Logic. It deals with reasoning that is approximate rather than fixed and exact. Fuzzy Logic has been extended to handle the concept of partial truth where the truth value may range between completely true and completely false. In fuzzy Logic, membership function represents the degree of truth. For any set, membership function lies between [0,1]. The purpose of introducing fuzzy logic in databases is to enhance the classical models such that uncertain and imprecise information can be represented and manipulated. This resulted in numerous contributions, mainly with respect to the popular relational model or to some related form of it.

2.3. Fuzzy Sets

The original interpretation of fuzzy sets arises from a generalization of the classic concept of a subset extended to embrace the description of "vague" and "imprecise" notions.

Definitions 2.3.1: A fuzzy set A over a universe of discourse X (a finite or infinite interval within which the fuzzy set can take a value) is a set of pairs

$$A = \{\mu A(x) / x : x \in X, \mu A(x) \in [0,1] \in \Re\}. \quad (2.1)$$

Where μA(x) is called the membership degree of the element x to the fuzzy set A. This degree ranges between the extremes 0 and 1 of the dominion the real numbers:

• μA(x) = 0 indicates that x in no way belongs to the fuzzy set A.

• μA(x) = 1 indicates that x completely belongs to the fuzzy set A.

Sometimes, instead of giving an exhaustive list of all the pairs that make up the set (discreet values), a definition is given for the function μA(x), referring to it as characteristic function or membership

The universe X may be called underlying universe

$$\mu A(x): X \rightarrow [0,1] \quad (2.2)$$

If the membership function produces only values of the set {0,1}, then the set that it generates is not fuzzy, but "crisp" (specific, exact, or precise).As mentioned previously, the universe of discourse X or the set of values being considered can be of two types:

• Finite or discreet universe of discourse X = {x1, x2,..., xn}, where a fuzzy set A can be represented by:

$$A = \mu 1 / x1 + \mu 2 / x2 + ... + \mu n / xn \quad (2.3)$$

• Infinite universe of discourse, where a fuzzy set A over X can be represented by:

$$A = \int \mu A(x)$$

2.4. Linguistic Labels

If an attribute is able of fuzzy treatment then linguistic labels can be defined on it. These labels will be preceded with the symbol $ to distinguish them easily. There are two types of labels and they will be used in different fuzzy attribute types:

(1) Labels for attributes with an ordered underlined fuzzy domain (Fuzzy Attributes Type 1 and 2) and (2) Labels for attributes with a non ordered fuzzy domain (Fuzzy Attributes Type 3 and 4).

Example 2.1: If we express the qualitative concept "young" by means of a fuzzy set, where the x-axis represents the universe of discourse "age" (in natural whole numbers) and the y-axis represents the membership degrees in the interval [0,1], then, following Equation 2.3, the fuzzy set that represents that concept could be expressed in the following way (considering a discreet universe):

Young = 1/0 + ... + 1/25 + 0.9/26 + 0.8/27 + 0.7/28 + 0.6/29 + 0.5/30 + ... + 0.1/34

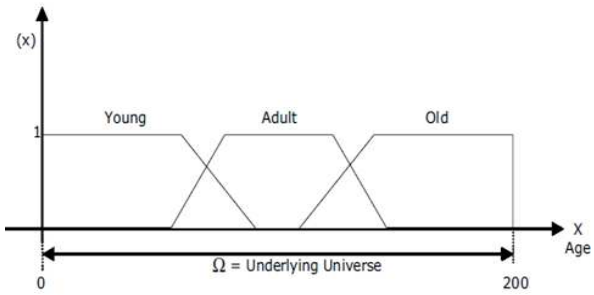

Figure 1. *Three linguistic labels of Example 2.1*

3. The GEFRED Model

The GEFRED model (GEneralised model Fuzzy heart Relational Database). It is based on the generalized fuzzy domain and generalized fuzzy relation, which include classic domains and classic.It is a possiblistic model, it particularly refers generalized fuzzy domains, but it also includes the case of where the underlying domain is not numeric but scalars of any type. It includes unknown, undefined and null values as well. The GEFRED model is based on the generalized fuzzy domain (D) and generalized fuzzy relation (R), which include classic domains and classic relations, respectively. To model the flexible queries and the concept of the fuzzy attributes, this paper uses an extension of SQL language called SQLf. We present in the following section this language.

3.1.1. Fuzzy Comparators

Besides the typical comparators (=,>, etc.), FSQL includes fuzzy comparators. There are many comparators, the most used are: FEQ (Fuzzy Equal than), (NFEQ: Necessarily FEQ),

FGT (Fuzzy Greater than), (NFGT: Necessarily FGT), FGEQ (Fuzzy Greater or Equal than), (NFGEQ: Necessarily FGEQ), FLT (Fuzzy Less Than), (NFLT: Necessarily FLT), FLEQ (Fuzzy Less or Equal than), (NFLEQ: Necessarily FLEQ), MGT (Much Greater Than), (NMGT: Necessity MGT), MLT (Much Less Than), (NMLT: Necessarily MLT), FINCL (Fuzzy INCLuded in), INCL, FDIF (Fuzzy DIFrent), (NFDIF: Necessary FDIF) [4, 10]. Like in SQL, fuzzy comparators compare one column with one constant or two columns of the same type.

3.1.2. Fuzzy Constants

Besides the typical constants (NULL), FSQL included many constants such as $[a,b,c,d], #n, $label, [n,m], UNKNOWN, UNDEFINED, etc.

3.1.3. Fuzzy Qualifiers

They are of two natures, absolute and relative [10].

3.1.4. Fuzzy Attributes

The classification adopted for the types of attributes is based on the approaches of representation and treatment of the "imprecise" data [9, 10]. These fuzzy attributes may be classified in four data types. This classification is performed taking into account the type of referential or underlying domain. In all of them the values Unknown, Undefined, and Null are included:

These attributes are of four types-

1.) Fuzzy Attributes type I: These attributes are represented as usual attributes because they do not allow fuzzy values. For example, queries of the kind, “Give me employees that earn a lot more than the minimum salary.”

2.) Fuzzy Attributes type II: This attribute stores the type of value corresponding to the data that we want to store, indicating its representation. It is an extension of the Type 1 that allows the storage of imprecise information, such as “he is approximately 2 meters tall.”

3.) Fuzzy Attributes type 3: In these attributes, some labels are defined that are scalars with a similarity relationship defined over them. For example, the value (1/dark, 0.4/brown), which expresses that a certain person is more likely to be dark than brown-haired but will definitely not be blond or ginger.

*4) Fuzzy Attributes types 4:*They are defined in the same way as Type 3 attributes, without it being necessary for a similarity relationship to exist between the labels (or values) of the domain. A possible example could be the type of role a client plays in a real estate agency, where the degree measures the importance with which a client is seeking or offering a property, without taking into account the similarity between the two roles.

3.1.5. Fuzzy Querying of Fuzzy Databases

Different frameworks for dealing with fuzzy querying of fuzzy databases exist. As mentioned above, Possibillistic Truth Values (PTV's) will be used in this paper. In that case, the evaluation of a criterion will lead to a PTV ~t (p) which has the advantage over regular satisfaction degrees s∈ [0, 1] of also being able to model an unknown satisfaction degree (p) = {(T, 1), (FM), or even a partially un-known satisfaction degree (e.g. i(p) = {(T,1), (F, 0.5)}. So, in case of unknown information in the database (which in a fuzzy databases is a possibility distribution over the domain of the attribute), this can be handled very easily using PTV's. On the other hand, inapplicable information, which in fuzzy databases still requires a special 'null' value, still can't be handled in a natural way, and if nothing else is done, will lead to a satisfaction degree expressed by the PTV {(T, 0), (F, 1)} (i.e. False'). Again, this is similar to the two situations above, and is not what is really desired when we want to deliver semantically correct answers to the user so, even when using PTV's, inapplicable information still requires a special approach. Again, it is proposed to use marked 'null' values, but this time only for the handling of inapplicable information because in a fuzzy database there is no need for a `null' value to express unknown information. As in the previous cases, an additional "mark" will be used to indicate to which domain value (including `unknown') the value 'inapplicable' should be treated semantically equivalent in case this value is queried.

4. Ontology

An Ontology is an explicit and formal specification of a conceptualization. Ontologies provides a shared understanding of a domain which allows interoperability between semantics. Components of an ontology:

- Terms
- Relations

The ontology that describes a Fuzzy Database Schema, as defined previously consists of a fuzzy Database schema and the fuzzy data stored in the Database (the tuples). This ontology, however, represent schemas and data simultaneously as ontology class cannot be instantiable twice, therefore two ontologies is defined, one of which describes fuzzy schemas as instances of a Database catalog ontology and the other which describes the same schema as a domain ontology which will allow the data instantiating it to be defined.

4.1. Ontologies vs. Databases

Ontologies allow representing knowledge of any domain in a formal and consensuated way. On the other hand, relational DBs also represent knowledge of any domain but following certain rules specified by ANSI Standard SQL. Nowadays, both technologies coexist together and they can exchange information to take advantage of the information that they represent. Several proposals have been developed for establishing the communication between ontologies and databases. Some of them consist of creating ontologies from a database structures, others populate ontologies using database data, and there are another which represent databases as ontologies.

4.1.1. Fuzzy Query Ontology

An ontology, called from now Fuzzy Query Ontology (FQO), represents all the basic fuzzy relational database query

constituents to get a flexible Select clause definition. After the instantiation of this ontology, the query process can start. This Ontology is specially designed for localhost application**.**

Database queries can be viewed as ontologies from two different perspectives: The first one, a query is a set of descriptions, conditions and rules defined as a set of instances of a Fuzzy Query ontology. In this sense, this query can be viewed as a SQLf statement where any element of the sentence is modeled in the ontology. The second one, a query is described as a a reduced domain ontology where a set of classes, properties and axioms represent a query specification and the ontology instances represent the resulting tuples.

Following fuzzy structures have been added to this ontology to complete the fuzzy RDBMS description:-

- Fuzzy Constraints: These restrictions, which are described in table can only be applied to fuzzy domains and are used either alone or in combination to generate domains such as, for example, not known, undefined, or null values are allowed, or only labels are allowed.
- Fuzzy Domains: These represent a set of values that can be used by one or more attributes. They are defined by a fuzzy data type, one or more fuzzy constraints, and those labels or discrete values that describe this domain.

4.1.2. Why Use Ontology?

The proposed Ontology, whose definition is extended in this paper, provides a frame where fuzzy data are defined in a platform-independent manner.

An implementation layer, which is responsible for parsing and translating user requests into the corresponding DB implementations in transparent is required to establish communication between the Ontology and the relational databases management system(RDBMS).

Fuzzy RDBMS requires complex data structures, in most cases, are dependent on the platform in which they are implemented. FRDB systems are poorly portable and scalable, even when implemented in standard RDBMs. The solution is to use Ontology which makes system independent of platform used. But all the previous work which uses Ontology are used for web application, none of the work is carried out for window application. Here, a new Ontology is proposed for the above problem.

5. Design & Implementation of Fuzzy Query Ontology

5.1. Fuzzy Query Executer Using Ontology

An implementation layer, which is responsible for parsing and translating user requests into the corresponding DB implementations in transparent, is required to establish communication between the Ontology and the relational databases management system (RDBMS). This implementation is designed with respect to corporate.

5.1.1. Parsing Technique Used

A parser generator which generates fully featured ob ject-oriented frameworks for building compilers, interpreters and other text parsers. In particular, generated frameworks include intuitive strictly-typed abstract syntax trees and tree walkers. SableCC [8] also keeps a clean separation between machine-generated code and user-written code which leads to a shorter development cycle.

An object-oriented framework that generates compilers (and interpreters) in the Java programming language. This framework is based on two fundamental design decisions. Firstly, the framework uses object-oriented techniques to automatically build a strictly-typed abstract syntax tree that matches the grammar of the compiled language and simplifies debugging.

Secondly, the framework generates tree-walker classes using an extended version of the visitor design pattern which enables the implementation of actions on the nodes of the abstract syntax tree using inheritance. These two design decisions lead to a tool that supports a shorter development cycle for constructing compilers

5.1.2. Fuzzy Query Translator

The queries are written in SQLf which is an extention to SQL. Fuzzy Query Executer works as a translator, that translates fuzzy queries to standard, SQL queries, and executes them with RDBMS.

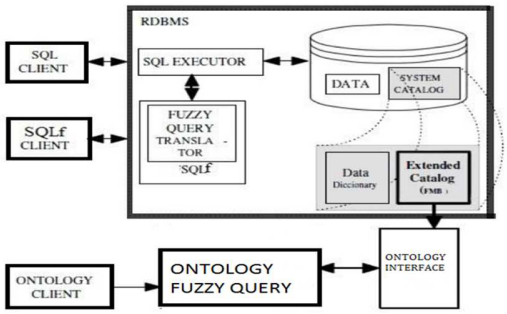

Figure 2. *Block Diagram of Fuzzy Query Ontology*

SQL Client directly interact with Relational Database Management System through SQL Executer and do not need any translation in between. SQLf is language used in the project tO retrieve data which is fuzzy in nature. SQLf Client need translation because Database can understand only SQL Language. So, Fuzzy Query Translator is used to convert SQLf query into SQL query. Ontology which is used in this project is independent of the platform used. Figure shows ontology is outside of RDBMS. Fuzzy Query Ontology is implemented in this project through various classes and its properties and behavior.A FMB is needed. In relational database,

The FMB will be responsible for organizing all the information related to the inexact nature or context of these attributes. The FMB is contemplated as an extension of the catalogue of the system (Data Dictionary),

FMB is created as shown in Block Diagram figure 2.

FMB contains few tables in database which stores the fuzzy attributes with their parameters to calculate fuzzy degree which handles the fuzzy database. This Block Diagram shows

how data is carried in database. An ontology for fuzzy information representation is proposed in this paper. The ontology includes knowledge about how to manage and represent uncertain and imprecise data. Figure 2 shows how the system catalog is related to the ontology modeling it. The Ontology Client module carries out the same operations through the Ontology than the DBMS Clients. The connection between the ontology and the database needs an interface, the Ontology Interface, which establishes the communication and refreshes the data. The fuzzy model representation comprises two well-differentiated parts. Firstly, the ontology must define the necessary classes and slots to represent the metadata. Secondly, the ontology will be able to represent classical or fuzzy information as instances of the relations. In this ontology, metadata establish how the fuzzy information will be stored.

5.2. Verification of Implemented Fuzzy Query Ontology

This project is implemented in context of some workers who are working for a various organization like Academics, government, Industry. The job expertise for organizations are AI, Statistics, Robotics, Expert System. There are various fuzzy attributes like age, salary, expertise and there are various linguistic labels for fuzzy attributes like age there are BABY, YOUNG, MIDDLE, MATURE, OLD and for salary labels are SMALL, MIDDLE, HIGH.

Examples:

1. Select name of worker, location of worker and age of worker from table name workers where fuzzy attribute age is ~young‘;

```
SQL_FQE>select fname,lname, age from workers where ~age is 'young';
+---------+------------+-----+--------------+
| fname   | lname      | age | fuzzy_degree |
+---------+------------+-----+--------------+
| neha    | ghaziabad  |  16 |  0.933333333 |
| mandeep | meerut     |  23 |  0.466666667 |
| rajesh  | muradnagar |  27 |  0.200000000 |
+---------+------------+-----+--------------+
3 rows in set
OK
SQL_FQE>
```

Figure 3. *shows the result of example 1*

2. Select name of worker, location of worker and salary of worker from table name workers where fuzzy attribute is salary is ~high‘;

```
SQL_FQE>select fname, lname, salary from workers where ~salary is 'high';
+---------+------------+--------+--------------+
| fname   | lname      | salary | fuzzy_degree |
+---------+------------+--------+--------------+
| ravikant | delhi     |   6500 |  1.000000000 |
| jogi    | delhi      |   6200 |  1.000000000 |
| rajesh  | muradnagar |   3800 |  0.520000000 |
| abhishek | bihar     |   3000 |  0.200000000 |
+---------+------------+--------+--------------+
4 rows in set
OK
SQL_FQE>
```

Figure 4. *shows the result of example 2*

3. Select fuzzy attribute age and function that calculate sum of salary from table called workers group by and order by fuzzy attribute age;

```
SQL_FQE> SELECT ~Age as age, FUZZY_SUM(Salary) FROM workers GROUP BY ~Age ORDER BY ~Age;
+--------+----------------+
| age    | sum            |
+--------+----------------+
| MATURE | 7473.333330100 |
| MIDDLE | 8266.666667800 |
| OLD    | 4333.333335500 |
| YOUNG  | 4026.666666600 |
+--------+----------------+
4 rows in set
OK
SQL_FQE>
```

Figure 5. *shows the result of example 3*

4. Select similarity label expertise from table name workers in order by Expertise only.

Expertise attribute is of type 3 fuzzy attribute having a similarity relation. This result shows the workers exact expertise and its related expertise with their fuzzy degree. In this

result mandeep and neha are having expertise AI that‘s why fuzzy degree is 1.0000 . rajesh is having expertise in Expert System which is 90% related to AI that is why fuzzy degree is 0.900000. Jogi is having expertise in Robotics which is 60% related to AI that is why 0.60000 is fuzzy degree.

```
Output - Fuzzy_Query_Executor (run)
SQL_FQE>select fname,#expertise as expertise from workers order by #expertise;
+----------+----------------+--------------+
| fname    | expertise      | fuzzy_degree |
+----------+----------------+--------------+
| mandeep  | AI             |  1.000000000 |
| neha     | AI             |  1.000000000 |
| rajesh   | AI             |  0.900000000 |
| jogi     | AI             |  0.600000000 |
| ravikant | AI             |  0.600000000 |
| abhishek | AI             |  0.200000000 |
| rajesh   | Expert Systems |  1.000000000 |
| mandeep  | Expert Systems |  0.900000000 |
| neha     | Expert Systems |  0.900000000 |
| jogi     | Expert Systems |  0.600000000 |
| ravikant | Expert Systems |  0.600000000 |
| abhishek | Expert Systems |  0.200000000 |
| ravikant | Robotics       |  1.000000000 |
| jogi     | Robotics       |  1.000000000 |
| neha     | Robotics       |  0.600000000 |
| rajesh   | Robotics       |  0.600000000 |
| mandeep  | Robotics       |  0.600000000 |
| abhishek | Robotics       |  0.200000000 |
| abhishek | Statistics     |  1.000000000 |
| neha     | Statistics     |  0.200000000 |
| mandeep  | Statistics     |  0.200000000 |
| rajesh   | Statistics     |  0.200000000 |
| ravikant | Statistics     |  0.200000000 |
| jogi     | Statistics     |  0.200000000 |
+----------+----------------+--------------+
24 rows in set
OK
SQL_FQE>
```

Figure 6. *shows the result of example 4*

5.3. Conclusion and Future Work

The above defined Ontology is specific to data retrieval from database. It is specifically designed for a flexible Select clause which represents all the basic fuzzy relational database query constituents. In this proposal of an ontology which represents a query structure regardless of any RDBMS implementation is defined. This ontology allows generating and executing any query on fuzzy or classical data according to the system where it is executed. The use of ontologies has provided of an independent layer where data can be modeled regardless of any RDBMS implementation particularity. Thus, the client interaction is performed through this ontology and an interface between the ontology and a real RDBMS is in charge of establishing the communication. A new Ontology is defined in this dissertation called Fuzzy Query Ontology.

A Fuzzy Query Ontology design, developed, implemented and verified successfully on given system environment (hardware & software) for data manipulation (data retrieval (select clause)).

References

[1] Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. and Lorensen, W. (1991). Object-oriented modeling and design. Englewood Cliffs, New Jersey: Prentice Hall.

[2] Carmen Martinez-Cruz.. "An Ontology to represent Queries in Fuzzy Relational Databases". IEEE 2011

[3] J. Galindo, A. Urrutia, and M. Piattini, "Fuzzy Database Modeling, Design and Implementation". Idea Group Publishing, 2006

[4] "Describing Fuzzy Database DB Schemas as Ontologies: A System Architecture View". Springer 2010. 13 th International Conference, IPMU july 2010.

[5] Sunita M. Mahajan, Vaishali P. Jadhav,"Analysis of Execution Plans in query optimization", International Journal of Scientific & Engineering Research, Volume 3, Issue 2, February-2012 , ISSN 2229-5518.

[6] Protégé, tool for creating and editing ontologies. http://protege.stanford.edu/, 2011.

[7] M. Neunerdt, B. Trevisan, T. C. Teixeira, R. Mathar, and E.- M. Jakobs, "Ontology-based corpus generation for web comment analysis," in ACM conference on Hypertext and hypermedia (HT 2011), (Eindhoven), 05 2011.

[8] Michaela Kreutzová, Jaroslav Porubän, "Automating User Actions on GUI: Defining a GUI Domain-Specific Language". In: CSE 2010: proceedings of International Scientific conference on Computer Science and Engineering: Stará Ľubovňa, Slovakia, 2010 pp. 60-67.

[9] Java Look & feel design guidelines, http://java.sun.com/products/jlf/ed2/book/, 2001

[10] J. Cheng, Z. M. Ma, and L. Yan, "f-SPARQL: a flexible extension of SPARQL," in Proceedings of the 21st international Conference on Database and expert systems applications: Part I, ser. DEXA'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 487–494.