Un Nuevo Paso hacia XQuery Flexible

A New Step towards Flexible XQuery

Marlene Goncalves, MSc., Leonid Tineo, PhD.
Universidad Simón Bolívar, Venezuela
{mgoncalves,leonid}@usb.ve

Recibido para revisión 24 de Septiembre de 2007, Aceptado 3 de Diciembre de 2007, Versión final 10 de Diciembre de 2007

Resumen—Actualmente, el Web es un gran repositorio de datos que requiere mecanismos de consulta adecuados. Existen muchos datos semi-estructurados publicados como documentos XML. En este contexto, XQuery ha sido concebido con el propósito de llegar a ser el estándar para consultas a bases de datos en XML. Este lenguaje de consulta está basado en el estándar de consulta XPath. XPath permite la formulación de criterios de selección booleana basados en expresiones de caminos. Por otra parte, existen una variedad de aplicaciones emergentes con requerimientos basados en preferencias que podrían no ser expresados con consultas basadas en lógica booleana. Una alternativa a este problema sería el uso de la lógica difusa. Distintos autores han aplicado lógica difusa a consultas a bases de datos relacionales. No obstante, pocos trabajos han sido presentados para hacer más flexible a los lenguajes de consulta sobre el Web. En este trabajo, se propone una extensión de expresiones de camino con lógica difusa, lo cual es un nuevo paso hacia un lenguaje XQuery más flexible para sistemas de información no tradicionales basados en el Web.

Palabras Clave—XML, XML Schema, XQuery, Lógica Difusa, SQLf.

Abstract—Currently, the Web is a great data repository that requires suitable querying mechanisms. There is a lot of published semi-structured data in XML documents. In this sense, XQuery has been conceived with the aim of becoming the standard for XML database querying. This query language is based on the single document querying standard XPath. It allows the formulation of Boolean selection criteria based on Path Expressions. On the other hand, there is a variety of emerging applications with preference based requirements that could not be expressed with Boolean logic based queries. An alternative way to this problem would be the use of fuzzy logic. Several authors have applied fuzzy Logic to relational database queries. Nevertheless, few works have been presented in order to make more flexible WWW query languages. In this work, we propose an extension of path expressions with fuzzy logic. It would be a new step towards a more flexible XQuery language for Web based non-traditional information systems.

Keywords—XML, XML Schema, XQuery, Fuzzy Logic, SQLf.

I. INTRODUCTION

WORLD Wide Web is a great data repository which requires declarative querying languages. In this sense, the World Wide Web Consortium (W3C) has made several works in order to provide standard languages for data exchanging and querying through Internet. A Lot of WWW published documents use the Extensible Markup Language XML [7], which it has been designed to describe data and it is a W3C recommendation for data exchanging. Data of XML documents may be described using XML Schema [27]. XML with a XML Schema is designed to be self-descriptive.

In order to provide querying capabilities over XML data, W3C have defined the language XQuery [29]. It is conceived to be for XML data like SQL is for relational databases. Currently, XQuery is supported by all the major database engines (IBM, Oracle, Microsoft, etc). This language will become a W3C standard; therefore, it will work among different products. XQuery is built on XPath [26] expressions that allow querying XML data. This language is based on the tree model for XML data. An XML document is seen in the XML data model [23] as a linearization of a tree structure. At every node in the tree there are several character strings. The tree structure and the character strings together form the information content of an XML document.

XPath expressions involve selection criteria that are expressed using Boolean logic. It leads to the same rigidity problem of classic database querying systems: interesting nearby answers may be lost and there is not discrimination over given answers [3][5]. Fuzzy sets have been proposed to be used in database querying in order to solve the rigidity problem of traditional querying systems. There are some different proposals of fuzzy set based querying languages. In particular, SQLf [3] is a fuzzy extension of SQL that allows fuzzy conditions in any place where SQL allows Boolean ones. Inspired in SQLf, we propose here an extension of XPath expressions as a step towards providing a more flexible

XQuery language. This language would allow to express user preferences. For example, it would be possible to find XML documents published in the WWW such as their recentness is "very current".

We have structured the present paper as follows: we review the related works in section 2 and the background in section 3; we present the syntax and semantics for fuzzy terms through XML Schema in section 4 and we propose fuzzy path expressions in section 5. Finally, we summarize and point out to the future in the section 6.

I. Related Works

Several research works have been devoted to provide more flexibility in database systems:

— OMRON [18] has a processor that contains a SQL extension with fuzzy logic and it is a fuzzy information retrieval library. It is a fuzzy query interface on traditional databases.

— FQUERY [17] is an effort that adds fuzzy query functionality over a small DBMS (MS-ACCESS). It allows using fuzzy quantifiers in order to qualify the quantity of satisfied criteria from a given list. Partitioned or nested queries are not allowed.

— ISKREOT (Intelligent System for Knowledge Representation using Expert system and Object Technology) [19] is a front-end intelligent information interface operating through a relational database ORACLE with fuzzy queries.

— FSQL [9][10], it is a fuzzy set based extension of SQL that incorporates some novelties to allow processing of inexact information. Many of fuzzy terms are predefined or must be defined in database modeling. It is in some way restrictive when a user wants to perform flexible querying over existing data. It is based on GEFRED [21] model.

— SQLf [3] is a flexible querying language for relational databases conceived to be a complete extension of SQL with fuzzy logic. Fuzzy queries supported by SQLf involve fuzzy terms (atomic predicates, modifiers, connectors, comparators and quantifiers) whose semantic depends of the user and the application domain. It supports the use of fuzzy quantifiers in grouping and nesting. SQLf has been extended to provide a fuzzy treatment of advanced features from the standards SQL2 [11] and SQL3[12].

— Additionally, Fukami and Umano [8] proposed a database engine with fuzzy querying capabilities based on fuzzy relational algebra operators. Wong and Leung in [31] and, more recently, Ma and Yan [20], evaluate a fuzzy query by means of translation to a SQL query but giving non-discriminated answers.

All these previous works are devoted to deal with relational databases, by means of fuzzy logic based extension of SQL. However, it would be useful to have flexible querying systems for XML. In this sense fewer works have been done.

Some ideas about how extend XPath with fuzzy terms have been presented in [4]. They have considered: providing a ranked list of retrieved information items rather than the usual set oriented one (Fuzzy Subtree Matching), specifying flexible selection conditions (Fuzzy Predicates) and allowing the specification of linguistic quantifiers as aggregation operators (Fuzzy Quantification). Nevertheless, they have not presented a way of specifying fuzzy terms but they introduced a set of few built-in predicates. On the other hand, they have kept out some interesting linguistic terms such as: modifiers, comparators and connectors. They see the ranking result of fuzzy criteria as an annotation (or comment) in the result. Combination of fuzzy predicates is made by means of arithmetic operations over ranking variables instead of using fuzzy logic operators.

Other interesting work in the way of making more flexible XML querying has been made by W3C in [2]. They presented a language designed to meet the Full-Text identified requirements. They propose to apply the notion of score to querying structured data. Besides specifying a match of a full-text search as a Boolean condition, full-text search applications typically also have the ability to associate scores with the results. Such scores express the relevance of those results to the full-text search conditions. In this later work, the notion of score is related to a variable that is returned by the query and is manipulated into the XQuery. Nonetheless, they propose the use of a generalized inexact match. They have not considered the possibility of leading user to specify its preferences by means of linguistic terms definition and combine it with fuzzy logic operators.

Finally, Soft Information Retrieval is another branch that uses fuzzy sets in Information Retrieval (IR). Information Retrieval [25] is a Computing Science branch that allows to store and access to a large amount of textual, visual, or auditory information. An Information Retrieval System (IRS) retrieves pertinent information to a user's query. The user's query, the representation of document contents and the matching from a query representation to a document representation may be uncertain and often vague.

There is a short survey of fuzzy approaches to IR in[6]. Analysis methods of natural language, probabilistic and fuzzy techniques are used to modeling the vagueness and uncertainty, which invariably characterize the management of information. Herrera et al. [15] present an IRS model that allows overcoming the problems of information loss and precision lack when working with discrete linguistic expression domains or when applying approximation operations in the symbolic aggregation methods. The work [16] introduces new averaging operators based on the weighted power mean for dealing with fuzzy information retrieval and avoiding that query results do not coincide with the intuition of the human being when handling "AND" and "OR" operations.

Previous work [1] deals about an IR system application based on a fuzzy database engine. Documents were registered in a catalog provided of theirs keywords annotated with relevance degrees. The system had also a thesaurus of keywords provided of a fuzzy similarity relation. Given a user requirement, retrieval was done by means of fuzzy deductive database techniques implemented in the underlying fuzzy database engine. We have also proposed and implemented data source selection systems in previous works [13][14]. Both systems allow document or data source selection according to user preferences about keywords and quality parameters such recentness, reliability, completeness, granularity and so on. They were conceived with a data source catalog. They provide a query by example interface over a SQLf engine. In [13] catalog may be provided in XML, nevertheless, it is loaded into a relational database for querying.

We would like to allow building Information Retrieval Systems, Data Source Selection Tools and other XML document based applications provided of fuzzy logic inside in a native way. In order to achieving this goal, we make here a new step towards providing a more flexible XQuery language. The scope this paper is to integrate fuzzy logic and XPath expressions, which will allow in future to extent more complex XQuery constructions that have their basis in XPath.

II. Background

Fuzzy sets [32] are intended to model vague concepts or classes. In a fuzzy set, each element is provided with a degree that represents its membership. These degrees induce an order that defines preferences. A function is used in order to represent such membership. The range of this function is the real interval [0,1], for a fuzzy set F it is denoted as μ_F.

Fuzzy set theory is the base of Fuzzy Logic. In this logic, the truth-value of a sentence (or satisfaction degree) $\mu(s)$ is in [0,1]. The value 0 represents completely false, 1 is completely true. This logic gives meaning to linguistic terms:

— Predicates that are atomic components of this logic defined by fuzzy sets.

— Modifiers: such as adverbs, negation and antonym, terms that allow defining modified fuzzy predicates by means of operations on the membership function.

— Comparators: kind of fuzzy predicates defined on pairs of elements, they establish fuzzy comparisons.

— Connectors: operators defined for combining fuzzy sentences. Fuzzy negation, conjunction and disjunction are extension of the classical. They preserve the existent correspondence with set operations minus, intersection and union, respectively.

— Quantifiers: terms describing quantities, such as “most of”, “about a half”, “around 20”. They are an extension of classical existential and universal.

One remarkable effort in flexible querying for relational databases is SQLf [3] a fuzzy extension of SQL. Fuzzy queries involve fuzzy terms whose semantic is user defined. SQLf basic querying structure is:

```
SELECT <attributes> FROM <relations>
WHERE <Fuzzy Condition>
WITH CALIBRATION [k|α|k,α];
```

The answer set of this query is the fuzzy set of rows with projected attributes of the SELECT clause in the Cartesian product of the relations in the FROM clause that satisfy the fuzzy condition in the WHERE clause. Fuzzy condition may involve user-defined terms, predefined operators and / or fuzzy sub queries. The (optional) WITH CALIBRATION clause indicates the best rows choice. Two kinds of calibration have been proposed:

— Quantitative calibration indicates the choice of top k answers, according to satisfaction degree.

— Qualitative calibration indicates the choice of answers whose membership value is greater or equal to the threshold α.

SQlf provides a kind of horizontal quantified queries [23] which general form is:

```
SELECT <attributes> FROM <relations>
WHERE Q(fc1,..,fcn)
WITH CALIBRATION [k|α|k,α];
```

Being Q a fuzzy quantifier, and $fc_1,..,fc_n$ a list of fuzzy conditions. This query returns the fuzzy relation Rf on $\{a \,/\, (\exists y \in R \,/\, y.A=a) \wedge (\mu(Q(X,fy)) \geq t)\}$, being the membership degree of each element a: $\mu_{Rf}(a) = \mu(Q(X,fy))$ (the truth degree of fuzzy quantified sentence *Q X’s are fy*), where $X=\{fc_1,..,fc_n\}$, fy is a fuzzy predicate on X whose satisfaction degree is $\mu_{fy}(fc)=\mu_{fc}(y)$ (the satisfaction degree of the row y to the fuzzy condition fc). The sentence *Q X’s are fy* is interpreted with the Yager’s decomposition interpretation [26][23]. The satisfaction degree of this sentence is $\mu(Q(X,fy)) = sup(min(\mu_Q(i),\mu_{fyi}))$ being μ_{fyi} the i-th higher value of the $\mu_{fy}(fc)$ degrees.

III. Fuzzy Terms

We would like to provide user with fuzzy querying capabilities in XPath expressions. In this sense, we must to allow the specification of user defined linguistic terms in order to be used in querying. In our fuzzy logic based extension, the allowed linguistic terms are predicates, modifiers, comparators, connectors and quantifiers. The general schema for fuzzy term specification is given in Table 1. We specify fuzzy terms through XML using the XML Schema language.

Table 1. XML SCHEMA FOR FUZZY TERMS

```
<xsd:complexType name="xsd:term">
   <xsd:all>
      <xsd:group ref="xsd:predicate"/>
      <xsd:group ref="xsd:modifier"/>
      <xsd:group ref="xsd:comparator"/>
      <xsd:group ref="xsd:connector"/>
      <xsd:group ref="xsd:quantifier"/>
   </xsd:all>
</xsd:complexType>
```

A. Fuzzy Predicates

The XML Schema for the fuzzy predicate definition is shown in Table 2. A fuzzy predicate is specified with a name, a domain and the membership function of the fuzzy set defining the predicate. This function that may be of three kinds: trapezium, extension and arithmetic expression.

Table 2. XML SCHEMA FOR FUZZY PREDICATE

```
<xsd:group name="xsd:predicate">
  <xsd:sequence>
   <xsd:attribute name="name" type="xsd:ID"/>
   <xsd:element name="domain"
      type="xsd:string"/>
   <xsd:choice>
      <xsd:group ref="xsd:trapezium"/>
      <xsd:group ref="xsd:single_extension"
         minOccurs="0" maxOccurs="unbounded"/>
      <xsd:element name="expression"
          type="xsd:string"/>
   </xsd:choice>
  </xsd:sequence>
</xsd:group>
```

A trapezium shape membership function is described by the sequence of its four inflection points x-values: x1, x2, x3 and x4. The specification of membership function in XML Schema is given in Table 3. The semantics is as follows: The first is where the function begins to growth form zero to one, the second is the first point in which the function raise the degree one, between these two points the satisfaction degree increases in straight line, between second and third point, the function gives the constant satisfaction degree one, the fourth point is where the element becomes completely excluded, between the third and the fourth, the satisfaction degree decreases proportionally.

Table 2. XML SCHEMA FOR TRAPEZIUM SHAPE MEMBERSHIP FUNCTION

```
<xsd:group name="xsd:trapezium">
  <xsd:sequence>
   <xsd:element name="x1" type="xsd:decimal"/>
   <xsd:element name="x2" type="xsd:decimal"/>
   <xsd:element name="x3" type="xsd:decimal"/>
   <xsd:element name="x4" type="xsd:decimal"/>
  </xsd:sequence>
</xsd:group>
```

A membership function defined by extension is specified giving a list of elements with their satisfaction degrees. Elements with membership zero are completely excluded of the fuzzy set; therefore, user may omit them in the extension specification. Any element out of the specified list is intended to have satisfaction degree zero. The specification of membership function in XML Schema is given in Table 4.

When the membership function is defined by an arithmetic expression, user just gives the expression using the predefined variable x intended for an element in the domain of the fuzzy predicate.

Table 4. XML SCHEMA FOR MEMBERSHIP FUNCTION DEFINED BY EXTENSION

```
<xsd:group name="xsd:single_extension">
  <xsd:sequence>
   <xsd:element name="value"
        type="xsd:string"/>
   <xsd:element name="degree">
        <xsd:simpleType>
           <xsd:restriction se="xsd:decimal">
               <xsd:minInclusive value="0"/>
               <xsd:maxInclusive value="1"/>
           </xsd:restriction>
        </xsd:simpleType>
   </xsd:element>
  </xsd:sequence>
</xsd:group>
```

Table 5 gives an example of user-defined fuzzy predicates with trapezium shape membership function.

Table 5. XML DOCUMENT EXAMPLE FOR FUZZY PREDICATES

```
<predicate name = "aprox_34_years">
   <domain> integer</domain>
   <trapezium> <x1> 30 </x1> <x2> 32 </x2>
               <x3> 36 </x3><x4> 38 </x4>
   </trapezium>
</predicate>
```

B. Fuzzy Modifiers

The XML Schema for a fuzzy modifier is shown in Table 6. This schema includes an identifier name attribute and an expression. This name and this expression define the fuzzy modifier. The expression tells how the modifier alters the degree of an element given a fuzzy predicate.

As an example, let's very be a fuzzy modifier that intensifies the application of a predicate. This modifier would be is defined as the square power of the degree. Table 7 contains a XML document for the definition of this modifier.

Table 6. XML SCHEMA FOR FUZZY MODIFIERS

```
<xsd:group name="xsd:modifier">
 <xsd:sequence>
  <xsd:attribute name="name" type="xsd:ID"/>
  <xsd:element name="expression"
     type="xsd:string"/>
 </xsd:sequence>
</xsd:group>
```

Table 7. XML DOCUMENT EXAMPLE FOR FUZZY MODIFIER "VERY"

```
<modifier name="very">
  <expression> power 2 </expression>
</modifier>
```

Table 8. XML SCHEMA FOR FUZZY COMPARATORS

```
<xsd:group name="xsd:comparator">
 <xsd:sequence>
  <xsd:attribute name="name" type="xsd:ID"/>
  <xsd:element name="domain"
     type="xsd:string"/>
  <xsd:choice>
     <xsd:sequence>
     <xsd:element name="type"><xsd:simpleType>
        <xsd:restriction base=xsd:string">
         <xsd:enumeration value="difference"/>
         <xsd:enumeration value="quotient"/>
        </xsd:restriction>
    </xsd:simpleType></xsd:element>
    <xsd:group ref="xsd:trapezium"/>
    </xsd:sequence>
    <xsd:element name="expression"
        type="xsd:string"/>
    <xsd:group ref="xsd:pair_extension"
        minOccurs="0"
        maxOccurs="unbounded"/>
  </xsd:choice>
 </xsd:sequence>
</xsd:group>
```

C. *Fuzzy Comparators*

We propose three kind of fuzzy comparators definition by:

— A fuzzy set over the quotient of compared numbers

— A fuzzy set over the difference of compared numbers

— Extension, indicating the satisfaction degree of related pairs.

The XML Schema for a fuzzy comparator is like that of Table 8. This definition contains a name attribute, the type of comparison and a type of membership function. Fuzzy comparators membership functions defined by extension are specified with the XML Schema of Table 9.

Table 9. XML SCHEMA FOR MEMBERSHIP FUNCTION BY EXTENSION ON PAIRS

```
<xsd:group name="xsd:pair_extension">
 <xsd:sequence>
  <xsd:element name="value1"
      type="xsd:string"/>
  <xsd:element name="value2"
      type="xsd:string"/>
  <xsd:element name="degree">
     <xsd:simpleType>
        <xsd:restriction base="xsd:decimal">
              <xsd:minInclusive value="0"/>
              <xsd:maxInclusive value="1"/>
        </xsd:restriction>
     </xsd:simpleType>
  </xsd:element>
 </xsd:sequence>
</xsd:group>
```

Let's see an example of fuzzy comparator definition. XML document of Table 10 defines the ">>>" comparator, intended as "much greater than" comparison, defined in terms of quotient membership degree to a fuzzy set.

Table 10. XML DOCUMENT DEFINING FUZZY COMPARATOR ">>>"

```
<comparator name=">>>">
  <type>quotient <type/>
  <trapezium>
    <x1> 1 </x1>  <x2> 10 </x2>
    <x3> infinit </x3> <x4> infinit </x4>
  </trapezium>
</comparator>
```

D. *Fuzzy Connectors*

Negation, conjunction and disjunction are built in connector. Their interpretations are complement to unit, max and the min, respectively. User may define own connectors. Table 11 shows the XML Schema for fuzzy connectors definition. It consists of an identifier name attribute and an expression. The expression tells how the connector combines the satisfaction degree of operands fuzzy conditions.

Table 12 exemplifies a connector definition: The "implication" connector defined by the logic equivalence $(A \rightarrow B) \equiv (\neg A \vee B)$, that is the expression "max(1-x,y)". Here x and y are interpreted as the satisfaction degree of left and right arguments of the connector, respectively.

Table 11. XML SCHEMA FOR CONNECTORS DEFINITION

```
<Xsd:group name="xsd:connector">
  <xsd:sequence>
   <xsd:attribute name="name" type="xsd:id"/>
   <xsd:element name="expression"
       type="xsd:string"/>
  </xsd:sequence>
</xsd:group>
```

Table 12. XML DOCUMENT EXAMPLE FOR FUZZY CONNECTOR "IMPLIES"

```
<connector name="implies">
  <expression>max(1-x,y)</expression>
</connector>
```

E. Fuzzy Quantifiers

The schema is given in Table 13 is intended for define a fuzzy quantifier. A fuzzy quantifier definition expressed in XML Schema contains a name attribute, the type of quantifier and a trapezium that defines the fuzzy quantifier.

Table 13. XML SCHEMA FOR DEFINITION OF FUZZY QUANTIFIERS

```
<xsd:group name="xsd:connector">
   <xsd:sequence>
      <xsd:attribute name="name"
      type="xsd:ID"/>
      <xsd:element name="type">
         <xsd:simpleType>
            <xsd:restriction base=xsd:string">
              <xsd:enumeration
              value="absolute"/>
              <xsd:enumeration
              value="proportional"/>
            </xsd:restriction>
         </xsd:simpleType>
      </xsd:element>
      <xsd:group ref="xsd:trapezium"/>
   </xsd:sequence >
</xsd:group>
```

Two types of fuzzy quantifiers are distinguished [24][30]:
— Absolute quantifiers represent amounts that are absolute in nature such as "about 5" or "more than 20". An absolute quantifier is represented by a fuzzy subset Q of real numbers.
— Proportional quantifiers are those as "at least half" or "most of" that are proportional in nature. They can be represented by fuzzy subsets of the unit interval, [0,1].

Table 14 presents a XML document for a user given definition of "at_least_3" absolute fuzzy quantifier.

Table 14. XML DOCUMENT WITH FUZZY QUANTIFIER "AT_LEAST_3"

```
<quantifier name="at_least_3">
  <type>absolute </type>
  <trapezium>
    <x1>0</x1> <x2>5</x2>
    <x3>infinit</x3> <x4>infinit</x4>
  </trapezium>
</quantifier >
```

IV. FUZZY PATH EXPRESSIONS

XQuery language uses path expressions in order to select nodes (or subsets) in XML document. These path expressions look very similar to the expressions you see when you work with a traditional computer file system [28]. For example, `author/address[@type='email']` expression returns authors where each of author addresses must have an attribute called "type" with the value "email".

Braga et al. [4] allow to use some built-in fuzzy predicates on attributes and tag names. In the tag name case, they suppose the existence of the tagname() expression that extracts the name of a tag in a XML document.

We propose to specify complex fuzzy conditions over tags and attributes. These conditions would involve user defined fuzzy terms as those presented in previous section. Hereafter we present these extensions by means of some representative examples. We do not go on details about XPath syntax, we assume reader to be familiar with it, otherwise see [26][28].

The result of a fuzzy path expression is a XML document where root nodes are presented in decreasing order of their satisfaction degree to the fuzzy condition. Such degrees are kept as special nodes attributes.

As an example, let's consider the XML document in Table XV and the following user requirement: "Search for employees with very high level studies". Let's assume that user define "`very`" fuzzy modifier as in Table 7 and "`high_level`" fuzzy predicate in Table 5. A fuzzy path expression representing this user requirement could be: `//employee[studies= very high_leve]`. This query returns the XML document in Table 16. We can see the satisfaction degree of the retrieved items obtained form the application of the modified fuzzy predicate. Only items with non zero satisfaction degree are retrieved.

Let's see another example. We search for employees in the XML document of Table 15 that meet at least three of the criteria: be young, tall, and heavy, of a high level study and with a regular salary. Want only elements over the threshold 0.5. This query is expressed as: `//employee [ at_least_3 ( age = young, height = tall, weight = heavy, studies = high_level, salary = regular ) ] with calibration 0.5`. In this case, at_least_3 is a user defined fuzzy quantifier (as in Table 14). Terms "young", "tall", "heavy", "high_level" and "regular" are user defined fuzzy predicates by means of XML in Table 5. Semantics of this query is the same of SQLf horizontal quatification, explained above. The result is shown in Table 17. Observe that retrieved items are listed in decreasing order of satisfaction degree.

Table 15. XML Document with Employees Information

```
<employee> <name> Cri Sto </name>
  <age> 33 </age><height> 200 </height>
  <weight> 77 </weight>
  <studies> 5 </studies><salary> 1500 </salary>
</employee>
<employee> <name> Kal Hil </name>
  <age> 33 </age><height> 183 </height>
  <weight> 92 </weight><studies> 1 </studies>
  <salary> 500 </salary>
</employee>
<employee> <name> Ken Cha </name>
  <age> 36 </age><height> 171 </height>
  <weight> 90 </weight>
  <studies> 4 </studies><salary> 1250 </salary>
</employee>
<employee> <name> Leo Tin </name>
  <age> 37 </age><height> 180 </height>
  <weight> 80 </weight><studies> 2 </studies>
  <salary> 850 </salary>
</employee>
<employee> <name> Rod Bin </name>
  <age> 34 </age><height> 192 </height>
  <weight> 120 </weight>
  <studies> 1 </studies><salary> 500 </salary>
</employee>
```

Table 16. XML Document with Fuzzy Employees Information

```
<employee degree=0.5625>
    <name> Cri Sto </name> <age> 33 </age>
    <height> 200 </height>
    <weight> 77 </weight>
    <studies> 5 </studies>
    <salary> 1500 </salary>
</employee>
<employee degree=0.2500>
    <name> Ken Cha </name> <age> 36 </age>
    <height> 171 </height>
    <weight> 90 </weight>
    <studies> 4 </studies>
    <salary> 1250 </salary>
</employee>
```

Table 17. XML Document with Fuzzy Employees Information

```
<employee degree=1.0>
    <name> Cri Sto </name> <age> 33 </age>
    <height> 200 </height>
    <weight> 77 </weight>
    <studies> 5 </studies>
    <salary> 1500 </salary>
</employee>
<employee degree=0.6>
    <name> Rod Bin </name> <age> 34 </age>
    <height> 192 </height>
    <weight> 120 </weight>
    <studies> 1 </studies>
    <salary> 500 </salary>
</employee>
<employee degree=0.5>
    <name> Ken Cha </name> <age> 36 </age>
    <height> 171 </height>
    <weight> 90 </weight>
    <studies> 4 </studies>
    <salary> 1250 </salary>
</employee>
```

V. Conclusion

Traditional querying languages suffer of rigidity, in the sense that fail in express user preferences and give discriminated answers. Despite expressive power XML querying languages, they also present this problem.

In this paper we have deal with the problem of giving more flexibility to XQuery by means of fuzzy logic use. As a step for the XQuery extension, we have focus our attention to XPath expressions. These expressions are the basis for XQuery. We have presented here the syntax and semantics of an XPath extension with fuzzy logic conditions involving user defined linguistic terms.

Main feature of our proposal is that it allows a large variety of fuzzy criteria expression using linguistic predicates, modifiers, comparators, connectors and quantifiers. We have presented XML Schema intended for these terms specification.

The result of a fuzzy path expression is a XML document where root nodes are presented in decreasing order of their satisfaction degree to the fuzzy condition. Such degrees are kept as special nodes attributes.

Next step would be the integration of satisfaction degrees in more complex queries that might be specified in XQuery. In a future work we will propose a fuzzy join and the use of fuzzy quantifiers in partitioning and nesting.

Finally, another interesting step would be to use fuzzy logic for answering ontology queries for the Semantic Web. We work at present time in this way. We hope in a near future to have a contribution in this field.

Acknowledgment

This work was supported in part by the Venezuela Foundation for Science, Innovation and Technology

FONACIT Grant G-2005000278.

There is a very special person that has inspired our lives and our works. Therefore, we would like to express a great acknowledgment to this person. All the things that we do, we do them by this person and without this person nothing we could do. This person is the King of Kings and Lord of Lords, the Savior for all human being: Jesus Christ.

REFERENCES

[1] Aguilera, A.; Subero, A.; Tineo, L. "Similarity–Based Queries for Information Retrieval". Lecture Notes in Computer Science, 1966, DNIS 2000, Pp. 148-156.

[2] S. Amer-Yahia, C. Botev, S.Buxton, P. Case, J. Doerre, D. McBeath, M. Rys, J. Shanmugasundaram, "XQuery 1.0 and XPath 2.0 Full-Text" W3C Working Draft 3 November 2005, www.w3.org/TR/xquery-full-text

[3] P. Bosc , O. Pivert, "SQLf: A Relational Database Language for Fuzzy Querying", IEEE Transactions on Fuzzy Systems, Vol 3, No. 1, Feb 1995

[4] D. Braga, A. Campi, E. Damiani, G. Pasi, PL. Lanzi, "FXPath: Flexible Querying of XML Documents", Proc. of EuroFuse 2002, Varenna, Italy, September 2002

[5] E. Cox "Relational Database Queries using Fuzzy Logic", Artificial Intelligent Expert, pp 23-29,Jan 1995.

[6] F. Crestani and G. Pasi, "Soft Information Retrieval: Applications of Fuzzy Set Theory and Neural Networks", Neuro-fuzzy tools and techniques, N.Kasabov Editor, Physica-Verlag, Springer-Verlag Group, pp. 287-313, 1999.

[7] Extensible Markup Language (XML). Available at http://www.w3.org/XML/. 2007

[8] S. Fukami and M. Umano. "Fuzzy Relational Algebra for Possibility-Distribution-Fuzzy-Relational Model of Fuzzy Data", Journal of Intelligent Information System, Vol 3, pp 7-27, 1994.

[9] J. Galindo, "New Characteristics in FSQL, a Fuzzy SQL for Fuzzy Databases". WSEAS Transactions on Information Science and Applications 2, Vol. 2, pp. 161-169, February 2005

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

[11] M. Goncalves and L. Tineo, "SQLf Flexible Querying Language Extension by means of the norm SQL2", The 10th IEEE International Conference on Fuzzy Systems, Vol 1, Dec 2001.

[12] M. Goncalves and L. Tineo, "SQLf3: An extension of SQLf with SQL3 features", The 10th IEEE International Conference on Fuzzy Systems, Vol 3, Dec 2001.

[13] M. Goncalves and L. Tineo, "A Web Tool for Web Document and Data Source Selection with SQLfi". Proceedings of the 9th International Conference on Enterprise Information Systems (ICEIS-2007). Madeira, Portugal. June 2007.

[14] M. Goncalves and L. Tineo, "The Egloo Fuzzy Web Data Source Selection Tool". Proceedings of DEXA 2007 Workshops, 2nd International Workshop on Flexible Database and Information System Technology - FlexDBIST . Regensburg, Germany. August 2007.

[15] E. Herrera-Viedma, A.G. López-Herrera, M. Luque and C. Porcel., "A Fuzzy Linguistic IRS Model Based on a 2-Tuple Fuzzy Linguistic Approach", International Journal of Uncertainty, Fuzziness and Knowledge-based Systems. In press, 2007.

[16] W-S. Hong, S-J. Chen, L-H. Wang, S-M. Chen, "A new approach for fuzzy information retrieval based on weighted power-mean averaging operators", Computers & Mathematics with Applications, v.53 n.12, p.1800-1819, June 2007.

[17] J. Kacpryzyk, S. Zadrozny, "Fuzzy Queries in Microsoft AccessTM v.2", Proc. of Fuzzy IEEE'95 Workshop on Fuzzy Database Systems and Information Retrieval, 1995.

[18] H. Nakajima, T. Sogoh, M. Arao, "Fuzzy Database Language and Library-Fuzzy Extension to SQL", Proc. of Second IEEE International Conference on Fuzzy Systems, 1983.

[19] K. Lee and G. Loo. "An Interface to Databases for Flexible Query Answering: A Fuzzy-Set Approach". Lecture Notes in Computer Science 1873. DEXA 2000.

[20] Z.M. Ma and Li Yan, "Generalization of Strategies for Fuzzy Query Translation in Classical Relational Databases. Information and Software Technology, Volume 49, Issue 2, February 2007.

[21] J. Medina, M. A. Vila, O. Pons. "GEFRED: A Generalized Model of Fuzzy Relational Databases," Informatizan Sciences, 1993

[22] H. Nakajima, T. Sogoh and M. Arao. "Fuzzy Database Language and Library-Fuzzy Extension to SQL", Proceedings of Second IEEE International Conference on Fuzzy Systems, pp 477-482, 1983.

[23] The XML Data Model. Available at http://www.w3.org/XML/Datamodel.html. 2005.

[24] L. Tineo, "A Contribution to Database Flexible Querying: Fuzzy Quantified Queries Evaluation", Doctoral Thesis, Universidad Simón Bolívar, Caracas, Venezuela, 2006.

[25] V. Rijsbergen, C.J. Information Retrieval. Butterworths, London, second edition, 1979.

[26] XML Path Language (XPath). Available at http://www.w3.org/TR/xpath20. 2007

[27] XML Schema. Available at http://www.w3.org/XML/Schema. 2007.

[28] XPath Introduction. Available at http://www.w3schools.com/xpath/xpath_intro.asp. 2007.

[29] XQuery 1.0: An XML Query Language. Available at http://www.w3.org/TR/xquery/. 2007

[30] R. Yager, Interpreting Linguistically Quantified Propositions, International Journal of Intelligent Systems, Vol. 9, (1994).

[31] M. Wong and K. Leung. "A fuzzy Database-Query Language". Information Systems. Vol 15. No. 5, pp 583-590, 1990.

[32] L.A. Zadeh, "Fuzzy sets". Information and Control 8, (1965).

Leonid Tineo (Caracas Venezuela, 1968). PhD in Computing, Universidad Simón Bolívar, Caracas, Venezuela, 2006. MsC in Computer Science, Universidad Simón Bolívar, Caracas, Venezuela, 1992. Eng. in Computing, Universidad Simón Bolívar, Caracas, Venezuela, 1990.

He is Titular Professor (since 2007), Staff Member of Universidad Simón Bolivar (since 1991). He has an Accreditation Venezuelan Investigator Promotion Program as Researcher Level 1 (since 2003). He has received the distinctions: Outstanding Professor CONABA (2002), Outstanding Educational Work USB (1999). He is the Coordinator of the Investigation and Development Group in Databases of the Universidad Simón Bolívar (since 2002). He has exerted the post of Information and Integration Coordinator of the Research and Development Deanship at Universidad Simón Bolívar (since 2002 until 2007). In the area of Fuzzy Databases, he has more than twenty articles in extenso in arbitrated Proceedings, more than fifteen published brief notes, tree papers in indexed journals, one book chapter and more than fifteen advisories of works conducing to academic titles. He is the responsible of the project "Creation and Application of Fuzzy Databases Management Systems" supported by FONACIT (since 2006).

Marlene Goncalves MsC in Computer Science, Universidad Simón Bolívar, Caracas, Venezuela, 2002. Lic. in Computing, Universidad Central de Venezuela, Caracas, Venezuela, 1998.

She is Aggregated Professor (since 2005), Staff Member of Universidad Simón Bolivar (since 2001). She has an Accreditation Venezuelan Investigator Promotion Program as Researcher Level 1 (since 2007). She has received the distinction Outstanding Professor CONABA (2002), She is the Chief the Investigation and Development Laboratory in Databases of the Universidad Simón Bolívar (since 2006). In the area of Preference Management for Databases, she has more than ten articles in extenso in arbitrated Proceedings, more than five published brief notes, tree papers in indexed journals and more than ten advisories of works conducing to academic titles. She is the co-responsible of the project "Creation and Application of Fuzzy Databases Management Systems" supported by FONACIT (since 2006).