Volume 2, Issue 5, May 2017 ISSN (online): 2456-0006

International Journal of Science Technology Management and Research

Available online at: www.ijstmr.com

Navigation on XML Schemas Using Innovative XSPath

Bhagyashree P. Khare
BE Computer
Shatabdi Institute of Engineering & Research
Nashik, India

Vidya A. Palde
BE Computer
Shatabdi Institute of Engineering & Research
Nashik, India

Komal S. Ghadge
BE Computer
Shatabdi Institute of Engineering & Research
Nashik, India

Vedanti R. Chavanke
BE Computer
Shatabdi Institute of Engineering & Research
Nashik, India

Suhas B. Gote
Asst. Professor
Shatabdi Institute of Engineering & Research
Nashik, India

Abstract: ***A Markup language has been mainly chosen for data representation, storage, and exchange in many different arenas to optimize the specific work. A query language need to progress both schema and path specification.Schemas are often used to constrain the content and structure of XML documents. That is by using previous one we can't navigate an XML document in an easy way and quickly. We propose a query language, named XSPath, specifically tailored for XML schema that works on logical graph-based representations of schemas, on which it enables the navigation, and allows the selection of nodes. However, XML Schemas are themselves XML documents. Thus, the structure of a schema can be navigated and its components can be retrieved through a path language. XSPath is a language tailored for specifying path expressions on XML Schemas. Also XQuery based translation proposed that can be exploited for the evaluation of queries.***

Keywords: XML schema, XPath, schema querying, XSPath query language.

I. INTRODUCTION

An XML document contains a prolog and a body. The prolog consists of an XML declaration, possibly followed by a document type declaration. The body is made up of a single root element, possibly with some comments and/or processing instructions. For instance, an XML document may be dynamically compiled from information contained in a database. XML schema is a description of a type of XML document, typically expressed in terms of constraints on the structure and content of documents of that type, above and beyond the basic syntactical constraints imposed by XML itself.

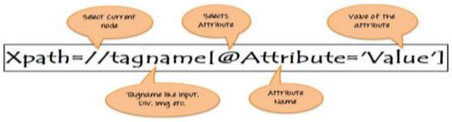


Fig.1.Examples of XML Input

These constraints are generally expressed using some combination of grammatical rules governing the order of elements, Boolean predicates that the content must satisfy, data types governing the content of elements and attributes, and more specialized rules such as uniqueness and referential integrity constraints. There are languages developed specifically to express.

II. LITERATURE SURVEY

S. Amer-Yahia, N. Koudas, and D. Srivastava in 2003 proposed that Schemas are regularly used to constrain the content material and structure of XML documents. this is by using using preceding one we are able navigate an XML record in an clean way and fast. We advocate a question language, named XSPath, specifically tailor-made for XML schema that works on logical graph-based representations of schemas, on which it permits the navigation, and allows the selection of nodes. however, XML Schemas are themselves XML files. consequently, the structure of a schema may be navigated and its components may be retrieved thru a route language. XSPath is a language tailor- made for specifying path expressions on XML Schemas. [7]

C.Y. Chan, W. Fan, and Y. Zeng in 2004 proposed that In Recent days exchange XML data more often in organizations and business sectors, so there is an increasing need for effective and efficient processing of queries on XML data. This paper presents a wide analysis to identify the efficiency of XML tree pattern matching algorithms. Previous years many methods have been proposed to match XML tree queries efficiently. In particularly TwigStack, OrderedTJ, TJFast and TreeMatch algorithms[3].

F. Cavalieri, G. Guerrini, and M. Mesiti in 2011 proposed that A Markup language has been specially selected for information representation, garge, and exchange in lots of unique area as to optimize the particular work. A question language want to progress each schemas and course specification. Generating X+ for extracting the xml attributes and factors from XML Schemas. A non-contraints development on XML route schema transformation provide a content retrieving technique like tool location. This language improvement device will increase the accuracy on repossessing paintings, it really works on tree representations of schema, on which it enables the navigation, and permits the selection of nodes and querying on schema. also XQuery primarily based translation proposed that can be exploited for the evaluation of queries. [8]

D. Colazzo and C. Sartiani in 2011 proposed that In now days data retrieval is the main focusing term in web data extraction. The process of XML data extraction in real time using search engines like Google, Ask, Bing and Yahoo etc. The RDBMS has some central methods to perform searching mechanism in real time data sets, but RDBMS is not suitable for XML data extraction. XQuery path language is the main methodology for Lowest Common Ancestors for implementing fuzzy type operations with XML data extraction. Fuzzy relational data extraction is very expensive of minimal cost[4].

Altova Ltd, XML Spy in 2012 proposed that An XML Schema describes the structure of an XML document. An XML document can be validated against an XML Schema to check whether it conforms to the requirements specified in the schema. If it does, it is said to be valid; otherwise it is invalid. XML Schemas enable document designers to specify the allowed structure and content of an XML document and to check whether an XML document is valid. The XML Spy interface is structured into three vertical areas. The central area provides you with multiple views of your XML document. The areas on either side of this central area contain windows that provide information, editing help, and

_le management features [5].

Cal`, G. Gottlob, G. Orsi, and A. Pieris in 2012 proposed that Schemas are often used to constrain the content and structure of XML documents. That is by using previous one we cant navigate an XML document in an easy way and quickly. We propose a query language, named XSPath, specifically tailored for XML schema that works on logical graph-based representations of schemas, on which it enables the navigation, and allows the selection of nodes. However, XML Schemas are themselves XML documents. Thus, the structure of a schema can be navigated and its components can be retrieved through a path language. XSPath is a language tailored for specifying path expressions on XML Schemas[6].

Federico Cavalieri, Giovanna Guerrini,and Marco Mesiti in 2014 proposed that Schemas are often used to constrain the content and structure of XML documents. They can be quite big and complex and, thus, difficult to be accessed manually. The ability to query a single schema, a collection of schemas or to retrieve schema components that meet certain structural constraints significantly eases schema management and is, thus, useful in many contexts. In this paper, we propose a query language, named XSPath, specifically tailored for XML schema that works on logical graph-based representations of schemas, on which it enables the navigation, and allows the selection of nodes. We also propose XPath/XQuery-based translations that can be exploited for the evaluation of XSPath queries. An extensive evaluation of the usability and efficiency of the proposed approach is presented within the EXup system [1].

III. EXISTING SYSTEM

XML markup language used to represent data on web. Schema documents (XSD) defines the rules to constrain the type and structure of the xml. XSD can be quite big and complex and thus, difficult to be accessed manually. The ability to query a single schema, a collection of schemas or to retrieve schema components that meet certain structural constraints is provided in XSPath query language. It is specifically tailored for XML schema that works on tree representations of schemas. The most important kind of expression in XSPath is a location path. A location path consists of a sequence of location steps. Each location step has axis, node test and predicates. An XS-Path expression is evaluated with respect to a context node. An Axis specifies the direction to navigate from the context node. The node test and the predicate are used to filter the nodes specified by the axis. It also provides XPath/XQuery-based translations that can be used for the evaluation of XSPath queries. An extensive evaluation of the usability and efficiency of the XS-Path is finally presented within the EXup system.

Disadvantages:

a) Dependency on path expression needs specification lead to expect flexible query language.
b) It failed to query on composition of schemas.
c) Complex schema reference files are not easily recovered.
d) An instance of application context in graphical work needs more schema advancement.

IV. PROPOSED SYSTEM

In this paper we propose a query language, named XSPath, specifically tailored for the retrieval of XML schema components. This language offers the ability of expressing retrieval needs on a logical representation of schemas, leaving aside the verbose XML schema syntax, thus greatly simplifying retrieval tasks, offering at the same time all the power and flexibility of a query language over graphical inspection tools. A key feature of the proposed language is that the expressions are specified on a two-level graph-based abstraction of schemas. These abstract representations make the specification of the expressions easier and leave to the language interpreter the burden of solving the gap between the logical (graph-based) and physical (textual) representations of schemas. The language captures a wide spectrum of retrieval needs in an XML schema that include: navigation on the nesting structure of element declarations, navigation on the type hierarchy generated by type restriction and extension, expression of conditions on cardinality, uniqueness and key constraints, types of element content and annotations. The paper also proposes approaches for the translation of XSPath expressions in XPath/XQuery expressions to be evaluated on the textual representations of the target schemas. The choice of implementing XSPath via translation is mainly due to the possibility of relying on XPath/ XQuery engines rather than developing a new engine specifically tailored for the evaluation of XSPath expressions.

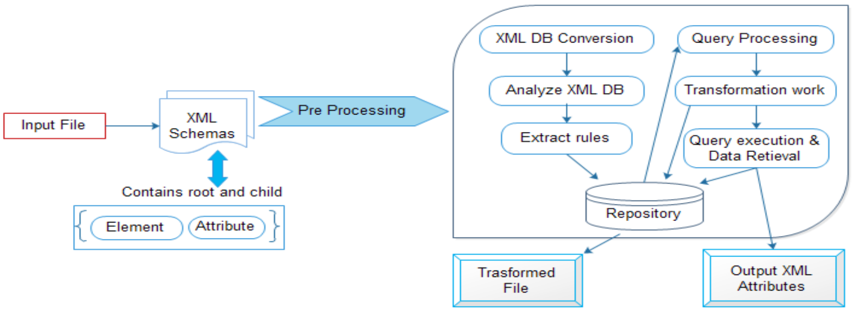


Fig. 2. Proposed System

Advantages:

a) A key feature of the proposed language is that the expressions are specified on a two-level graph based abstraction of schemas.

b) This language offers the ability of expressing retrieval needs on a logical representation of schemas, leaving aside the verbose XML schema syntax, thus greatly simplifying retrieval tasks, offering at the same time all the power and flexibility of a query language over graphical inspection tools.

V. SYSTEM ARCHITECTURE

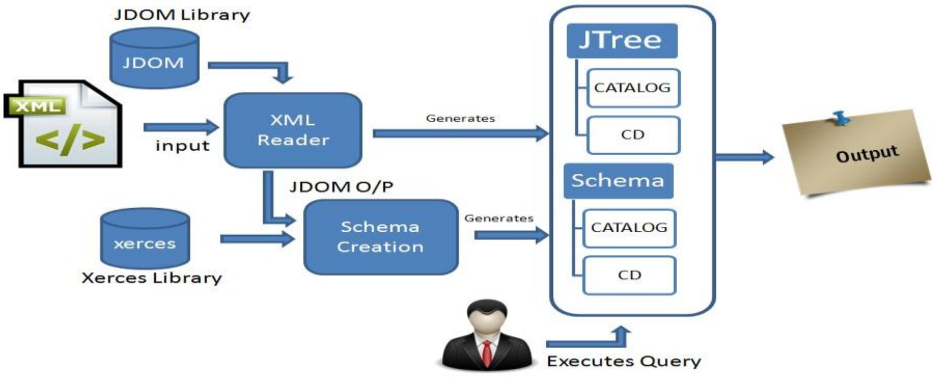


Fig. 3. System Architecture

VI. MODULE DESCRIPTION:

There are four modules used as follows,

1. XML Pre-Processing
2. Query translation
3. User Module.
4. XSPath Type System Module.

1. XML Pre-Processing

In this model,user gives XML construction XML which is utilized to recognize the given setting hub. From these we need to decide the Axis,node selector and indicator. In the wake of deciding assess characteristic for every hub. Assessing hubs we store this data in database.

2. Query translation

In this model,take the yield from pre-processor database. Assessment of preprocessor database is accomplished for metadata. Client ought to take question to assess the preprocessor database, after assessment of question is done from the client which is then parsed into tokens. At the point when Metadata is conrmed assessment is accomplished for every token. Change over the token which keep running on database made amid pre-preparing, then run this inquiry on pre-processor database.

3. User Module

In this model, clients are having verification and security to get to the points of interest which is introduced in the framework. Before getting to or seeking the points of interest client ought to have the record in that else they ought to enroll first.

4. XML Type System Module

XSPath Type System exhibits the arrangement of XSPath writing rules, which decide the sorts of the hubs that can be identified by a XSPath expression. The guidelines depends on a setting sorts T that signifies the sorts of the hubs on which an expression can be assessed. The sort of the first venture of an outright XSPath expression is resolved.

VII. TECHNICAL SPECIFICATION

i. Hardware Requirement

- System : Pentium
- Hard Disk : 30GB
- Monitor : VGA color
- MEMORY : 1GB

ii. Software Requirement

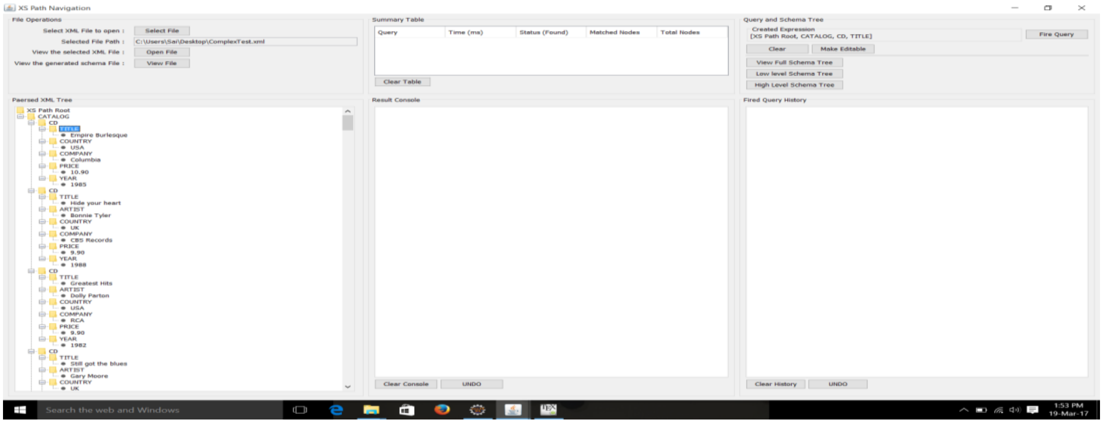
- Eclipse Mars.1 Release (4.5.1)
- Java 1.8.0_65
- W3c-dom
- xbean-2.2.0
- xerces-2.0.2

VIII. ALGORITHM

Algorithm Steps:

1) Step 1: Start.
2) Step 2: Select xml file.
3) Step 3: If user wants to open file.
4) Step 4: Select node from parsed tree to create expression.
5) Step 5: Fire query to see selected nodes..

IX. EXPERIMENTAL RESULTS AND ANALYSIS



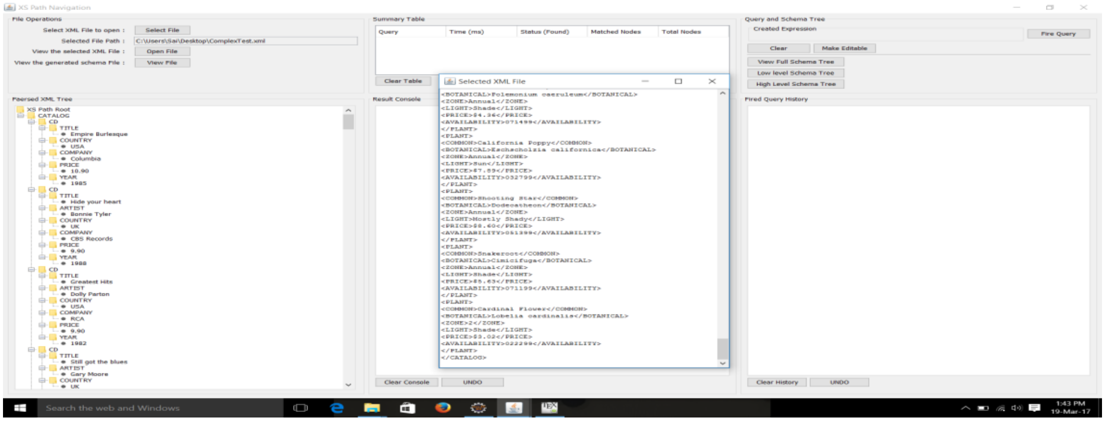
Selected File Path : C:\Users\Sai\Desktop\ComplexTest.xml

Query	Time (ms)	Status (Found)	Matched Nodes	Total Nodes

Created Expression
[XS Path Root, CATALOG, CD, TITLE]

```
XS Path Root
  CATALOG
    CD
      TITLE
        Empire Burlesque
      COUNTRY
        USA
      COMPANY
        Columbia
      PRICE
        10.90
      YEAR
        1985
    CD
      TITLE
        Hide your heart
      ARTIST
        Bonnie Tyler
      COUNTRY
        UK
      COMPANY
        CBS Records
      PRICE
        9.90
      YEAR
        1988
    CD
      TITLE
        Greatest Hits
      ARTIST
        Dolly Parton
      COUNTRY
        USA
      COMPANY
        RCA
      PRICE
        9.90
      YEAR
        1982
    CD
      TITLE
        Still got the blues
      ARTIST
        Gary Moore
      COUNTRY
        UK
```

In above figure, User can select xml file from select xml file button and path will be generated. After selecting xml file, parsed tree will be generated.

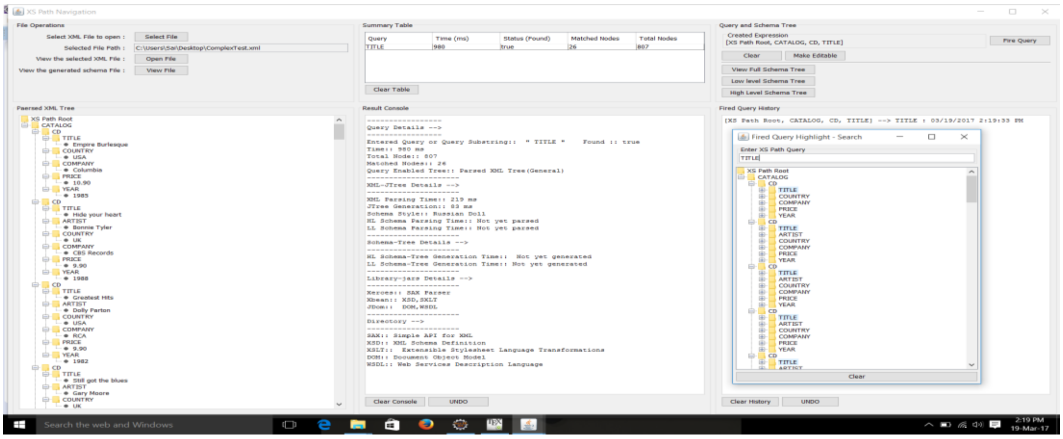


Selected File Path : C:\Users\Sai\Desktop\ComplexTest.xml

Query	Time (ms)	Status (Found)	Matched Nodes	Total Nodes

```
<BOTANICAL>Polemonium caeruleum</BOTANICAL>
<ZONE>Annual</ZONE>
<LIGHT>Shade</LIGHT>
<PRICE>$4.36</PRICE>
<AVAILABILITY>071499</AVAILABILITY>
</PLANT>
<PLANT>
<COMMON>California Poppy</COMMON>
<BOTANICAL>Eschscholzia californica</BOTANICAL>
<ZONE>Annual</ZONE>
<LIGHT>Sun</LIGHT>
<PRICE>$7.89</PRICE>
<AVAILABILITY>032799</AVAILABILITY>
</PLANT>
<PLANT>
<COMMON>Shooting Star</COMMON>
<BOTANICAL>Dodecatheon</BOTANICAL>
<ZONE>Annual</ZONE>
<LIGHT>Mostly Shady</LIGHT>
<PRICE>$8.60</PRICE>
<AVAILABILITY>051399</AVAILABILITY>
</PLANT>
<PLANT>
<COMMON>Snakeroot</COMMON>
<BOTANICAL>Cimicifuga</BOTANICAL>
<ZONE>Annual</ZONE>
<LIGHT>Shade</LIGHT>
<PRICE>$5.63</PRICE>
<AVAILABILITY>071199</AVAILABILITY>
</PLANT>
<PLANT>
<COMMON>Cardinal Flower</COMMON>
<BOTANICAL>Lobelia cardinalis</BOTANICAL>
<ZONE>2</ZONE>
<LIGHT>Shade</LIGHT>
<PRICE>$3.02</PRICE>
<AVAILABILITY>022299</AVAILABILITY>
</PLANT>
</CATALOG>
```

In above figure, the generated xml file is viewed.



Selected File Path : C:\Users\Sai\Desktop\ComplexTest.xml

Query	Time (ms)	Status (Found)	Matched Nodes	Total Nodes
TITLE	980	true	26	807

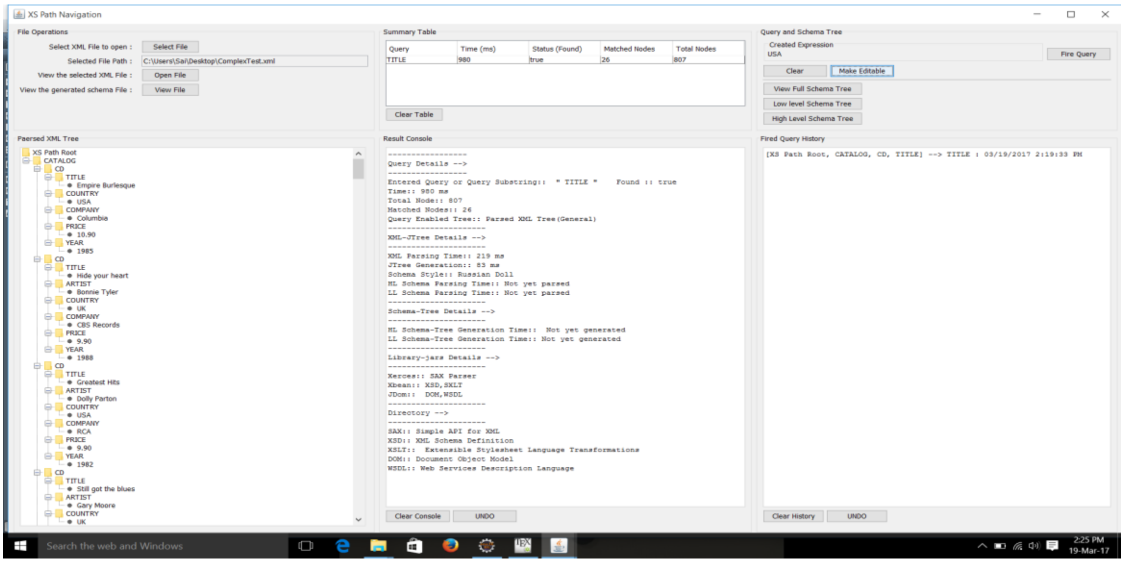
Created Expression
[XS Path Root, CATALOG, CD, TITLE]

```
-----------------
Query Details -->
-----------------
Entered Query or Query Substring::  " TITLE "    Found :: true
Time:: 980 ms
Total Node:: 807
Matched Nodes:: 26
Query Enabled Tree:: Parsed XML Tree(General)
---------------------
XML-JTree Details -->
---------------------
XML Parsing Time:: 219 ms
JTree Generation:: 83 ms
Schema Style:: Russian Doll
ML Schema Parsing Time:: Not yet parsed
LL Schema Parsing Time:: Not yet parsed
---------------------
Schema-Tree Details -->
---------------------
ML Schema-Tree Generation Time::  Not yet generated
LL Schema-Tree Generation Time:: Not yet generated
---------------------
Library-jars Details -->
---------------------
Xerces:: SAX Parser
Xbean:: XSD,SXLT
JDom::  DOM,WSDL
---------------------
Directory -->
---------------------
SAX:: Simple API for XML
XSD:: XML Schema Definition
XSLT::  Extensible Stylesheet Language Transformations
DOM:: Document Object Model
WSDL:: Web Services Description Language
```

```
[XS Path Root, CATALOG, CD, TITLE] --> TITLE : 03/19/2017 2:19:33 PM
```

Enter XS Path Query: TITLE

In above figure, the node is selected and query is fired.



Selected File Path : C:\Users\Sai\Desktop\ComplexTest.xml

Query	Time (ms)	Status (Found)	Matched Nodes	Total Nodes
TITLE	980	true	26	807

Created Expression
USA

```
-----------------
Query Details -->
-----------------
Entered Query or Query Substring::  " TITLE "    Found :: true
Time:: 980 ms
Total Node:: 807
Matched Nodes:: 26
Query Enabled Tree:: Parsed XML Tree(General)
---------------------
XML-JTree Details -->
---------------------
XML Parsing Time:: 219 ms
JTree Generation:: 83 ms
Schema Style:: Russian Doll
ML Schema Parsing Time:: Not yet parsed
LL Schema Parsing Time:: Not yet parsed
---------------------
Schema-Tree Details -->
---------------------
ML Schema-Tree Generation Time::  Not yet generated
LL Schema-Tree Generation Time:: Not yet generated
---------------------
Library-jars Details -->
---------------------
Xerces:: SAX Parser
Xbean:: XSD,SXLT
JDom::  DOM,WSDL
---------------------
Directory -->
---------------------
SAX:: Simple API for XML
XSD:: XML Schema Definition
XSLT::  Extensible Stylesheet Language Transformations
DOM:: Document Object Model
WSDL:: Web Services Description Language
```

```
[XS Path Root, CATALOG, CD, TITLE] --> TITLE : 03/19/2017 2:19:33 PM
```

In above figure, there is option to fire query using make editable.

X. APPLICATION

1. XML Schema is successful in that it has been widely adopted and largely achieves what it set out to, it has been the subject of a great deal of severe criticism.
2. XML files are using in all type of domains application for documentary process.
3. Usability and the efficiency of the translation have been increased due to its dynamic processing framework.
4. Retrieving schema process is simple than the existing XPATH navigating process.
5. From the experimental work we can definitely give that the transforming and parsing time will be too low.

By Developing this Xml query language specifically personalized for the evaluation of typical expressions easily

CONCLUSION AND FUTURE WORK

In this project, we have proposed a navigational XML schema query language. The language has been defined, by specifying its syntax and semantics, and a translation process has been proposed to evaluate expressions in the language through existing XPath / XQuery engines. The XPath was designed to operate on schema file as schemas are the structural representation of XML file. But the performance of XPath was limited in some context of rules. Hence proposed a navigational XML schema query language which defines a type analysis for the given language as well it optimizes schema conversion with translation algorithm independently. Hence XSPath is been derived from XPath language. The proposed system gives the conversion of schema file to simplified tree format structure which will helps us to compare performance of both the languages. XPath translation algorithm is introduced to define correctness and complexity of result conversion. In this way Ex-up is an optimized technique for translation and evaluation of XSPath expression. The specified performance analysis using different datasets, shows the usability of the given system and its practical applicability.

REFERENCE

1. M Speretta and S Gauch, "Personalized Search Based on User Search Histories", Proceeding Of International Conference on Web Intelligence, pp. 622-628, 2005.
2. F Liu, C Yu and W Meng, "Personalized Web Search for Improving Retrieval Effectiveness", IEEE Transactions On Knowledge And Data Engineering, pp. 28-40,Volume 16, 2004.
3. C Liang, "User Pro_le for Personalized Web Search", International Conference on Fuzzy Systems and Knowledge Discovery, pp. 1847- 0, 2011.
4. X Pan, Z Wang and X Gu, "Context-Based Adaptive Personalized Web Search for Improving Information Retrieval Effectiveness", International Conference on Wire-less Communications, Networking and Mobile Computing, pp. 5427 - 5430, 2007.
5. K.W.T. Leung, D.L. Lee and Wang-Chien Lee, "Personalized Web search with location preferences", IEEE 26th International Conference on Data Engineering, pp.70 I - 712, 2010.
6. O. Sha_q, R. Alhajj and 1. G. Rokne, "Community Aware Personalized Web search", International Conference on Advances in Social Networks Analysis and Mining, pp.3351 - 355,2010.
7. Li, L. Yang, Z. and Kitsuregawa, M. (2009) Rank Optimization of Personalized Search. Journal of the Database Society of Japan, 8, pp.35-40.
8. Page, L., et al., The PageRank Citation Ranking: Bringing Order to the Web, Technical Report, (1999).
9. Brin, S., R. Motwani, and T. Winograd, What can you do with a web in your pocket, Data Engineering Bulet in, 21(2), (1998), pp.37-47.
10. Bharat, K. and G.A. Mihaila, when experts agree: using non-a_liated experts to rank popular topics . ACM Trans. Inf.Syst., 20(1), (2002), pp.47-58.
11. Haveliwala, T.H., et al., Evaluating strategies for similarity search on the web, in Proceedings of the 11th nternationalconference on World Wide Web, ACM: Hon- olulu, Hawaii,USA, (2002), pp. 432 - 442.