Journal of Advances and Scholarly Researches in Allied Education
Vol. 15, Issue No. 12, December-2018, ISSN 2230-7540

An Imperative Role of Xml Parsing and Xml Query Language

Swati Gupta[1]* Dr. Ashish Chourasia[2]

[1] Research Scholar, University of Technology, Jaipur, Rajasthan

[2] University of Technology, Jaipur, Rajasthan

Abstract – Increasing the use of internet technology, the XML markup language attaches remarkable importance due to its language equality and freedom in the use of data exchange and data transfer by process of the web environment. In order to improve the processing efficiency of XML parser, a method needs to be found to get obtain least processing time when parsing XML documents. XML is capable of extracting data from an XML document with no information or understanding of its contents. To achieve this clarity, XML documents must comply with XML requirements. Memorizes interpreted XML documents as byte sequence and retrieves prior parsing outcomes whenever the byte array of a current XML document somewhat matches the recited sequences. In this thesis we will study on XML Parser and XML- query languages which provide graphical illustration of such knowledge and translate it that is in-memory structure for the entire function to be used.

Keywords – XML, XML Document, Parser, Query Languages

----------X----------

1.1 INTRODUCTION

XML Parser where parser is a bit of program which provide graphical illustration of such knowledge and translate it that is in-memory structure for the entire function to be used. Parsers are used in applications everywhere. A XML Parser is a parser which deliberate to peruse XML and produce a path to utilize XML[3] for projects. Several kinds of things, and each ha has its points of interest. Unless a program duplicates the entire XML record as a part, each program should run or approach an XML parser. DOM is sustain navigate and transformed XML documents

- Hierarchical tree illustration of document
- Tree pursue typical API
- Creating tree is vendor precise

DOM is a language-neutral pattern

- Obligatory subsist for Java, C++, CORBA, JavaScript, C# -
- Could switch to a different language.

The DOM is an interface oriented API which permit for navigation of the complete document as else tree of node objects signify the document's contented. A DOM document get generated to parser, and it produce physically by client (by restriction).

1.2 XML

XML is the meta-language characterized by the World Wide Web Consortium (W3C) which get utilized to portray an expansive scope of progressive increase language. It is a lot of regulation, rules, and shows for portraying organized information in a simple book, editable record. Utilizing a book group rather than a paired arrangement permits the developer or still end client to take a gander or use the information without depending on the program which delivered it. Anyway the essential maker and buyer of XML data of PC program and not the end-client. Like HTML, XML utilizes labels and traits. Labels are words sectioned by the " characters and traits are strings of the structure 'name="value"' that are within labels. While HTML determines what each tag and property implies, just as their introduction properties in a program, XML utilizes labels just to delimit bits of data and leaves the understanding of the data to the application that utilizations it.

As it were, XML characterizes just the structure of the archive and doesn't characterize any of the introduction semantics of that record. Improvement of XML began in 1996 prompting a W3C Recommendation in Febuary of 1998. Notwithstanding, the innovation isn't totally new. It depends on SGML, created in the mid 1980's and turned into an ISO standard in 1986. SGML was broadly utilized for huge credentials ventures and have enormous network which as knowledged

working to SGML. The planners of XML got the greatest pieces of SGML, utilized their experience such as direct and delivered an innovation which has similarly as amazing SGML, yet greatly simplier and simpler to utilize. XML-based documents might utilized in a extensive assortment of applications with perpendicular markets, web based trade, business-to-business correspondence, and endeavor function informing.

XML is rising as a de facto typical for data swap over internet. It is derived from SGML and HTML. The current infrastructure is available to deal with HTML content and the same infrastructure can be re-used to work with XML; hence most of the companies can easily integrate their business document. For example, General Motors [4], Jet Blue Airline use XML in their business.

Apart from these characteristics, the XML has additional characteristics and these are: - high flexibility, platform independency, portability, simplicity, and usability. It is a human language which is conversable and readable by the people who had no formal introduction to XML. Flexibility means programmer or user can create his own tags; which are not limited to standard tags; which are not predefined by programmer. XML is vendor independent and system independent also. If a program is using vendor dependent tags, there are limitations such as the browsers and other programs associated with it. In such a case, these tags need to be approved of by the concerned authorities. Even the users have to get accustomed to the usage. This is a time consuming process. This limitation is overcome by XML. It is fully compatible with applications like JAVA, C etc and it can be combined with any other application which is capable of processing XML, irrespective of the platform. Due to all these characteristics, today XML is in use in most of the businesses.

XML was emerged from SGML [5]. In 1980, SGML (Standard Generalized Markup Language) was defined by ISO 8879. SGML has been the standard, vendor-independent way to maintain repositories of structured documentation for more than decade, but it was not well suited to serve documents over the web. In 1990 HTML came into picture. It is simple and has predefined tag semantics and tag sets, but unable to represent structural relationship and search utility. These drawbacks are overcome by XML. It provides a facility to define tags and structural relationships between them. There is no predefining meaning of tag set and preconceive semantics. It will either define by the applications that process them or by style sheet this enhances the functionality of the internet. Following fig 1.1 summarizes XML development.

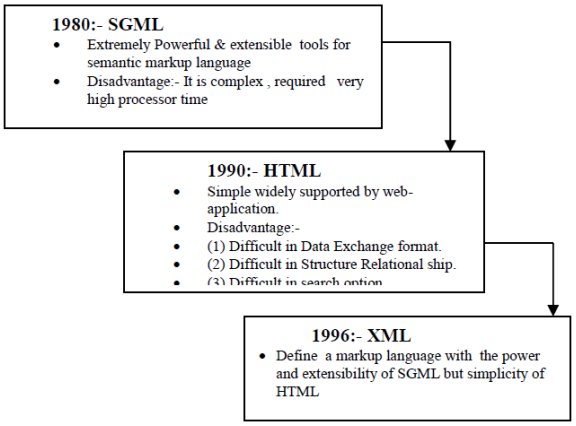

Fig 1.1: XML Progression

The XML file is simple text file which is composed of element node, attribute node and comment node.

1.3 XML DATA REPRESENTATION COMPONENTS

The XML element is a piece of text that is enclosed within a start tag (<-- /) and end tag (/ -- >) It represents the object or entity. Element node is simple element, complex element and mixed element. An XML element can be associated with one or more XML attributes. The attribute node represents the characteristics of element node. There are two major issues to be considered when using attributes. An attribute cannot be repeated more than once within a single element definition and its value part must be enclosed in double-quotes as a string regardless its data-type. Three special-purposes attributes namely “id”, “idref” and “idrefs” are used to support references between different parts of an XML documents. The benefit of this layout (1) minimizing the XML document size by preventing data redundancy (2) reducing the cost of updating such reference data by ensuring that update will take place at one position in the XML document. Comment node represents comments of the XML document. It can be found anywhere inside XML documents and they are ignored by most of the API that manipulates XML documents. The XML document structure is mapped in fig 1.2 as follows:-

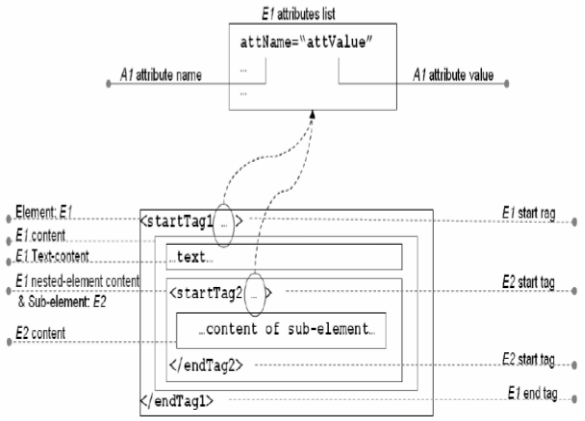

Fig 1.2 XML File structure

Other XML components are DTD (Documents Type Definition) and XML schema. They are used for validating XML document.

This XML file is converted into predefined data representation format which is called as XML data model. XML data is represented into different data model, some of the important data models are: (1) XPath (2) DOM (3) XML infoset (4) edge label tree (5) node label tree (6) OEM etc. This model has an important role in query processing. The desired data must be retrieved as per the requirement.

1.4 XML- QUERY LANGUAGES

XML supports two types of queries: IR Style and Database Style of query. Database style of queries is used on 'Data-Centric' application.

The Database queries are broadly classified into containment queries, order queries and content-based queries [6]. The containment queries retrieve 'Parent-Child', 'Ancestor-Descendent' relationships. The order sensitive queries are forward and backward navigation queries. They are following, following-sibling, preceding, preceding-sibling, child axis, self axis queries. The content-based queries are associated with element or attribute node content .These query check the attribute and element content and retrieve it.

These queries are further classified into: tree structure queries and starting node queries. The tree structure queries return a small tree. They are further classified into simple path query which corresponds to a chain-path and branch path expressions – which corresponds to a small tree, called twig. The starting node queries are further divided into "total matching queries and partial matching queries. The total matching queries start from the root of the document where as partial matching queries start from some internal node.

IR-Style of queries is utilized to query text-dense XML repository that worth factor are involving lengthy text. These types of queries do not work well in common database query standards. These queries are classified into DB+IR Style and IR-Style of queries. DB+IR queries improve database-style XML queries like XPath and XQuery queries. For IR-Style of queries, a full-fledged retrieval technique is required, for example 'keyword' search is used for it. Some of the query language supports IR Style of query, or Database Style of query or both.

XML has a rich set of query languages. Milestone of XML languages [7] are listed on the following tables 1.1:-

Table 1.1: XML Query Languages

Query Language	Language Type	Input Model	Class of Query	Public Recognition
Lorel[49]	Declarative	OEM	Path expressions within OQL	1997
XML-QL[48]	Functional	Node-Labeled Tree Data Model	Pattern Matching	1998
XQL[50]	Functional	Node-Labeled Tree Data Model	XQL based on path expressions	1999
Quilt[51]	Functional	Edge - Labeled Tree Data Model	Quilt Expression	2000
XPath[16]	Functional	XPath	Axis query and predicate query	2002
XQuey[15]	Functional	XPath	FLOWR expression	2007

XPath and XQuery [8] are two major languages for database-style of queries and initially urbanized and optional by the W3C association [9]. XPath [10] is a fundamental query language which chooses the nodes from XML documents from the root pathway. By using XPath query, the XML tree is traverse in different direction. XQuery [11] language is additional significant than XPath. An XQuery include axis traversal of XML tree as well as For-Let-Where-Return (FLWR) clauses, that may nested and collected by complete majority [12] means each clause includes a sub-XQuery. The XPath is used for traversal of a complete XML file in different directions like forward, backward direction. There is lots of work is performed on the query optimization of the XML file. XQuery not only traverse a complete tree but also performed selective operation on it. It performs almost all type of query operation of XML database. Much expressive powered of XQuery increased the optimization and evaluation complexity. In literature different XQuery optimizer techniques are proposed by tree algebra for XML (TAX) [13] generalized tree pattern (GTP) [14], and tree logical class (TLC) [15] etc. For our research we select XQuery language and all different types of query, especially FLOWR clause which may return a single node, twig pattern and all order node i.e. following, following-sibling, preceding, preceding-sibling etc. Another reason of selection of XQuery Language, it satisfies the need of all community i.e. "Document Community ", "Database Community" and "Programming Community".

- Document-Centric applications often require text search, XQuery supports operations on document order and axis expressions, which are used to navigate in the document and to access the context of particular document fragments.

- Data-Centric applications often require very efficient selection, retrieval, and transformation of small fragments of data stored in massively large databases. XQuery incorporates features from query languages for relational databases (SQL) and Object-Oriented databases (OQL).

XQuery is a purely functional language which supports user defined functions and recursion.

The different feature of XQuery Language as per specification of [W3C3, W3CS4] is:-

- It is an XML query language with some programming language features and SQL-like semantics.

- It is developed from XPath data model.

- It has all functionalities, libraries and capabilities of XPath2.0 (i.e. XPath 2.0 is a subset of XQuery)

- It is supported by most commercial RDBMS such as IBM, Oracle and Microsoft SQL-Server.

- In addition to XPath2.0 capabilities, XQuery supports FLWOR [W3CS4] expressions (FLWOR is an acronym for "For, Let, Where, Order by, Return")

- XQuery1.0, a W3C Recommendation on 23/Jan/2007, is the latest version of XQuery.

Hence the researcher used the XQuery language for the research the detail of XML query processing is discusses as following:-

1.5 XML QUERY PROCESSING

As per W3C [8] XML processes are divided into two steps: - "External Query is processing "and "Internal Query processing "[16]. The fig 1.3 provides the abstract view of XML query processing.

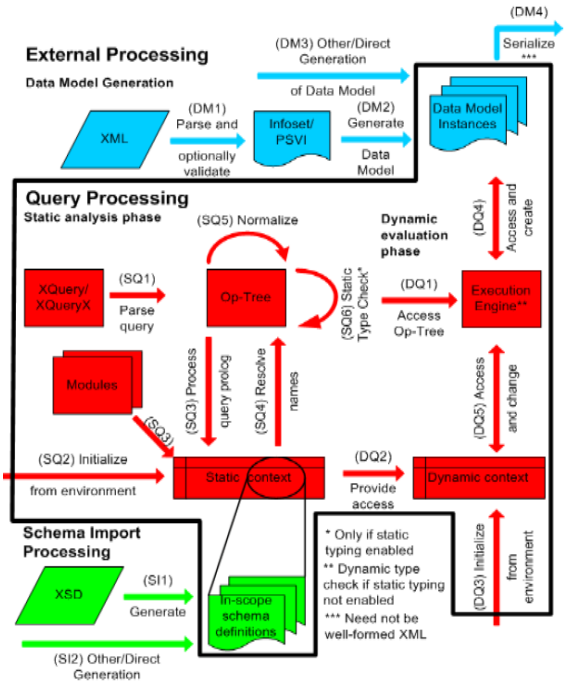

Fig 1.3: XML Query Processing

External Query Processing: - In these steps, the XML document is converted into data model. This data model is used for query processing. It is a two step process:-

Step I: - Data Model Generation: - XML file is parsed. Different tokens are generated and are validated against elements and attributes declarations and target namespace. It generates information set called as PSVI (Post-Schema Validation Infoset). From PVSI XML-data model is generated.

Step II: - In scope schema define: - XDM i.e. XML Schema Definition Model is used for defining schema definition. It defines all schema used by XML file. The file is divided into three parts: - in-scope schema types, in-scope element definition and in-scope-attribute declaration. 'In-scope schema type' includes definition of all types of definitions found in imported schema. 'In-scope element declaration' includes all element declarations found in imported schema. 'In-scope attribute declaration' defines all the attributes found in imported schema.

Internal Query Processing: - In this step XQuery is processed against XML file which is stored into XML Data model**.** Following steps are used for query processing:-

1. **Parsing**: - The query is parsed and checked for grammatical error, i.e. syntax error. If no error is found, an internal operation tree of the parsed query is created.

2. **Normalization or OP-Tree Generation: -** In this process, complex query is rewritten into

simple query expression i.e. converting the query expression into core language. If the query is not well-typed, static type errors are generated. For instance, a comparison between an integer value and a string value might be detected as a type error during the static type analysis.

For instance, the following [expression/query]

for $i in (1, 2),

$j in (3, 4)

Return

Element pair { ($i,$j)

is normalized to the Core expression

For $i in (1, 2) return

for $j in (3, 4) return

element pair { ($i,$j)

3. **Static type analysis**: - This is an optional step. Static type analysis checks whether each query is well-typed, and if so, determines its static type. Static type analysis is defined only for query Core language. Static type analysis works by recursive application of the static typing rules over a given expression.

4. **Dynamic Processing:-**This step is divided into two sub steps.

   - Dynamic Context Processing: - The dynamic semantics of query depends on the dynamic input. Dynamic input defines processing environment.

   - Dynamic Evaluation: - In this phase, dynamic value of a query is computed. All dynamic values are provided in this phase. It may be intermediate result of the operation, or reading the value of variable at runtime, dynamic variable. This Dynamic evaluation may result in a value OR a dynamic error, which may be a non-type error or a type error.

In XML Query Processes, important task is XML Data Model generation i.e. Organizing XML data file in such a way that desired data can be accessed easily.

1.6 DIFFERENT APPROACHES OF XML STORAGE AND QUERY PROCESSING TECHNIQUES

XML data is represented into three types of file: data-centric file, document-centric file and mix-mode file which is combination of data-centric and document-centric file. The stored information is retrieved by using different types of queries. Two types of queries are used on these file: IR queries and data centric queries. Information Retrial (IR) style of queries are used by document centric file. The database styles of queries are used by data centric file. Therefore, the different strategies are used for retrieving desired data e.g. for IR style of queries keyword search algorithm and ranking results are major issues while desire data retrieval with minimum query processing time is major challenge for database queries. For data centric query, different types of indexing techniques are applied on these file and have different query processing algorithm. The index strategies are classified as on residential type, structural relationships between different XML elements. The residential indexes are classified as main memory based index and disk-based index. The main memory based indexes are resided into computer's main memory and disk-based index are reside on secondary memory. The main memory style of index defines how to store entire documents into main memory. These types of index have excellent response time for structural-join queries because they avoid expensive cost of I/O operations. Its accomplishment is simple and sufficient for quarrying diminutive XML document and inefficient for large XML document. Because loading whole XML file from derivative memory interested in main memory increased high I/O cost, the need for disk-based algorithms is required. Now a day memory is not a problem thus disk base indexes can use memory as per their requirement but reduce query processing time is a major challenge for these indexes. Alternatively, XML indexing algorithms can be evaluated on the basis of how the XML structural relationship is encoded in the index file [17] .These indexes map the document property like edge structure, values and their node relationships in the index using physical and logical address. These properties are mapped as per the query work load. Thus two parameters need to be considered while designing this index i.e. flexibility and selectivity. Flexibility means index structure such that it can handle any arbitrary query work load. And selectivity means instead of indexing entire document create index on document fragments which are frequently queried by the users or application. Like RDBMS technology XML also have secondary index for has reducing index size and accelerates query performance. The secondary indexes are well established in relational database but it is still not well developed in XML, because XML data is hierarchical and semi-structure data, creating index is a major challenge on it. In current years, important attempt was dedicated for increasing high-performance XML database system by designing different style of index on document property like value index on content node value, structural relationship on XML elements, by creating unique identification of each XML node and on different feature of XML document. For the study

indexing techniques are classified into two types: (1) XML-RDBMS approach (2) Native Approach

CONCLUSION

XML defines a cross platform data format and Java provides a standard cross platform programming platform. Together, XML and Java technologies allow programmers to apply Write Once, Run Anywhere™ fundamentals to the processing of data and documents generated by both Java based programs and non-Java based programs. The utilize of structured documents in XML has a wide range of fields of use. Much of the time, when runtime is appropriate and the execution window is unmistakably limited, it is important to process documents of a significant size. Broadly, XML indexes are classified into two types: 'Value Indexes' and 'Structural Indexes'. Value indexes are built on XML data values i.e. from value node. The structural indexes are built on structure of XML documents.

REFERENCES

[1]. Extensible Markup Language, http://www.w3.org/TR/REC-xml.

[2]. OASIS, http://www.oasis-open.org.

[3]. Juancarlo Anez (1999). "Java XML Parsers-A Comparative Evaluation of 7 Free Tools," Java Report Online.

[4]. Yi Chen, Susan B. Davidson, Yifeng Zheng (2009). "A bi-labeling based XPath processing system" Information Systems 35, 2010, doi: 10.1016/j.is.2009.05. ACM

[5]. W3C Website. Extensible Markup Language (XML) 1.0 (Fourth Edition). Online:http://www.w3.org/TR/REC-xml/, Accessed on: 30/10/2006

[6]. Barbara Catania and Anna Maddalena, Athena Vakali (2005). "XML Document Indexes: A Classification " SEPTEMBER • OCTOBER 2005 Published by the IEEE Computer Society 1089-7801/05/$20.00 © 2005 IEEE IEEE INTERNET COMPUTING

[7]. Mikael Fernandus Simalango (2008). "XML Query Processing and Query Languages: A Survey" Property of Amikelive.com – Technical Paper Series 25/10/2008

[8]. Byron Choi, Mary Fernandez, Jerome Simeon (2002). The XQuery Formal Semantics: A Foundation for Implementation and Optimization May 31, 2002

[9]. W3C Consortium, http://www.w3.org, 2006

[10]. W3C Consortium, XML Path Language (XPath) 2.0, http://www.w3.org/TR/xpath20/, 2006.

[11]. W3C Consortiu (2006). XQuery 1.0: An XML Query Language, http://www.w3.org/TR/xquery/.

[12]. D.D. Chamberlin (2002). "XQuery: An XML Query Language," IBM Systems J., Vol. 41, No. 4.

[13]. H. Jagadish, S. Al-Khalifa, A. Chapman, L. Lakshmanan, A. Nierman, S. Paparizos, J. Patel, D. Srivastava, N. Wiwatwattana, Y. Wu, and C. Yu (2002). TIMBER: A Native XML Database. The VLDB Journal- Volume 11, pages 274-291.

[14]. Byron Choi, Mary Fernandez, Jerome Simeon (2002). The XQuery Formal Semantics: A Foundation for Implementation and Optimization May 31, 2002.

[15]. Z. Chen, H.V. Jagadish, L.V.S. Lakshmanan, and S. Paparizos (2003). "From Tree Patterns to Generalized Tree Patterns: On Efficient Evaluation of XQuery," Proc. 29th Int'l Conf. Very Large Data Bases (VLDB '03).

[16]. S. Paparizos, Y. Wu, L.V.S. Lakshmanan, and H.V. Jagadish (2004). "Tree Logical Classes for Efficient Evaluation of XQuery," Proc. 23rd ACM SIGMOD Int'l Conf. Management of Data (SIGMOD '04).

[17]. Mohammed Al-Badawi, Dr. Siobhán North, Dr. Barry Eaglestone (2007). Research memorandum Indexing XML Databases: Classifications, Problems Identification and a New Approach, 15th November, 2007 in The University of Sheffield Department of Computer Science.

Corresponding Author

Swati Gupta*

Research Scholar, University of Technology, Jaipur, Rajasthan