Building XML-based Website Using eXist Native XML Database Platform

Prasetiyo
SarMag Program of Information Technology
Faculty of Industrial Technology
Gunadarma University, Jakarta, Indonesia
prasetiyo@student.gunadarma.ac.id

Adang Suhendra
Information Technology Department
Faculty of Industrial Technology
Gunadarma University, Jakarta, Indonesia
adang@staff.gunadarma.ac.id

Abstract

As volumes of XML data grow, we need to store them in effective way. One solution to it is in native XML database. This paper presents an exploration and implementation of XML document retrieval using native XML database tool:eXist. Before we exploit some functionality it has, we build a simple wrapper tool to extract information from HTML to XML. After we have XML document, we begin our experiment to build simple site. The site is developed under eXist platform to make use all of eXist ability and also to implement some of XML-related technology functionality. The main purpose of this paper is to demonstrate eXist as native XML database platform and recommend XML as database format.

Keywords: eXist, native XML database, wrapper, XML storage, XQuery, XPath, XSLT

1. Introduction

The emergence of XML as a new document format offers a lot of advantages. Some advantages that many people get are their interoperability, platform-independent, and easiness to transform to other format as needed. Moreover, today XML has been broadly used as data storage, like database as well. Its close relation with database format in the structured manner and its free defined tag makes XML as an alternative format for database. However, the role of XML as database format is relatively new compared with another format like SQL, which has broadly used and well-known. XML is still promising a lot of opportunities to be continuously developed.

Not like other document and database format such as Microsoft Word and SQL, which are binary and proprietary, XML is ASCII (or plain-text) file without proprietary data. It makes XML is easily to be read and accessed by any platform without considering compatibility issue and dependency to specific software application. Therefore, XML does not need underlying process that makes it flexible and less overhead.

As the volume of XML-based information increases, the need for a query language to efficiently query and make use of that XML data is obvious [4]. It makes W3C, as a consortium that develop many standards related to WWW, release some recommendations related to information retrieval in XML data. Some of their recommendations are XQuery and XPath that will explain in more detail later. Such recommendations encourage XML to have an ability just like another RDBMS, i.e. SQL to access its data.

This paper presents an exploration and implementation of XML document retrieval using native XML database tool:eXist. This paper is organized as follows. Section 2 explains XML-related technology: XQuery and XPath, that support XML as database and a glimpse of XSLT to transform XML to the intended format. In section 3, eXist as tool that used in this paper is described in more detail. Section 4 presents our implementation by using eXist. Section 5 presents result and discussion. Section 6 concludes the paper.

2. XML-related Technology

As a number of data in XML document grow and become abundant, the way to retrieve those data is needed. XML is a structured and well-organized document, so it is easy to retrieve information on it. At the first, there is no standard way to retrieve data in XML document. W3C as a consortium which has developed many standard for internet, begins to develop new recommendation to provide new way to access data in XML document. Then, the new recommendation that supports XMLs ability as database emerged: XPath and XQuery. The ability of both is only in accessing data in XML document. To make use XPath and XQuery result, we need to transform to another format. Then, other XML-related technology is needed: XSLT. In this section, we will explore the three.
2.1. XPath

XML Path Language (XPath) Version 1.0 was endorsed as a W3C Recommendation in November 1999. XPath allows us to address parts of an XML document. XPath method of addressing document components, with its basic facilities for text string, number, and boolean (true-false) manipulations, allows for document searches down to the node level. Thus, it provides more precision in targeting and retrieving data[8].

XPath not only has ability to access XML document based on name or certain condition, but it can also use element relationship. In XPath, such relationship is called axes. Figure 1 shows several XPath axes.

<table>
<thead>
<tr>
<th>Specifier</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>child</td>
<td>Specifies all direct children of the current node. A child node is nested directly within the current node.</td>
</tr>
<tr>
<td>descendant</td>
<td>Specifies descendants of the current node. Includes child nodes, all children of child nodes, their children, and so on.</td>
</tr>
<tr>
<td>parent</td>
<td>Specifies the direct parent of the current node.</td>
</tr>
<tr>
<td>ancestor</td>
<td>Specifies all ancestor of the current node, including the parent node, parents of parents, and so on.</td>
</tr>
<tr>
<td>following-sibling</td>
<td>All nodes that follow the current node and have the same parent element.</td>
</tr>
<tr>
<td>preceding-sibling</td>
<td>All nodes that precede the current node and have the same parent element.</td>
</tr>
<tr>
<td>following</td>
<td>All nodes that follow the current node, regardless of nesting level.</td>
</tr>
<tr>
<td>preceding</td>
<td>All nodes that precede the current node, regardless of nesting level.</td>
</tr>
<tr>
<td>self</td>
<td>Specifies the current node.</td>
</tr>
</tbody>
</table>

Figure 1. XPath axes[5]

In general, XPath is used as anchor in accessing XML document. Logically, XPath can be described as individual or group of element selection based on given parameter. The result of the selection is used by another XML technology, like XQuery as given parameter to continue retrieving information. For more information about XPath, please visit http://www.w3schools.com/XPath/xpath_functions.asp.

2.2. XQuery

XML is a versatile markup language, capable of labeling the information content of diverse data sources including structured and semi-structured documents, relational databases, and object repositories. A query language that uses the structure of XML intelligently can express queries across all these kinds of data, whether physically stored in XML or viewed as XML via middleware. This specification describes a query language called XQuery, which is designed to be broadly applicable across many types of XML data sources [7].

The best way to explain XQuery is to say that XQuery is to XML what SQL is to database tables [1]. As the name XQuery is a language for querying XML data. It uses XPath expression to determine accessed element. XQuery became W3C recommendation in January 2007. It is really a new technology and does not broadly used as query language for XML document. On the other hand, XQuery has a great ability compared to SQL as long as XML became broadly used as database format.

One simple case that XQuery can solve is “Give me all countries in Asia” in the following XML document:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<countries>
  <country>
    <name>Australia</name>
    <capitol_city>Canberra</capitol_city>
    <continent>Australia</continent>
  </country>
  <country>
    <name>Japan</name>
    <capitol_city>Tokyo</capitol_city>
    <continent>Asia</continent>
  </country>
  <country>
    <name>China</name>
    <capitol_city>Beijing</capitol_city>
    <continent>Asia</continent>
  </country>
  <country>
    <name>Indonesia</name>
    <capitol_city>Jakarta</capitol_city>
    <continent>Asia</continent>
  </country>
  <country>
    <name>Mexico</name>
    <capitol_city>MexicoCity</capitol_city>
    <continent>America</continent>
  </country>
</countries>
```
With XQuery syntax as follow:
for $i in doc("countries.xml")/countries/country
where $i/continent="Asia"
order by $i/name
return data($i/name)

Then the result is:
China
Indonesia
Japan

From the example, we can conclude that XQuery syntax is similar to SQL query. It means there are many things that XQuery can do, and it must be easy to work with as easy as SQL. For more information about XQuery definition, please visit: http://www.w3schools.com/xquery/xquery_reference.asp

2.3. XSLT

We can use both XPath and XQuery to access data in XML document. They are not able to deliver the result to easy readable by human or computer application. We need another XML-related technology to make the result valuable, so we must transform or format them to deliver some meanings.

XSLT (or only XSL) stands for XML Stylesheet Language Transformation. In essence, XSL is two languages, not one. The first is a transformation language, the second a formatting language. The transformation language is useful independently of the formatting language[6]. With XSLT, we can transform XQuery result to any format. Whether to present in web browser as HTML, shown in handheld device as WML, or as printable format as XML-FO. Because of its formatting ability, it can collaborates with CSS or XSLT itself.

XSLT also make use XPath as its navigator to determine retrieved information. So, the three XML-related technology can be used together. We can determine information that we want to get in XQuery with XPath, then format the result with XSLT. In this paper, we only use XSLT as transformation tool to present XQuery result to HTML format.

3. eXist: The Native XML Database Tool

The need of specific tool that can handle XML as database has become important as many technology supported emerge. The most popular database management system, mySQL also has ability to handle XML document but it is not built in pure XML nature. One database tool that supports the three XML-related technologies described above is eXist (or eXist-db).

eXist-db is an open source database management system entirely built on XML technology. It stores XML data according to the XML data model and features efficient, index-based XQuery processing. eXist-db provides a powerful environment for the development of web applications based on XQuery and related standards. Entire web applications can be written in XQuery, using XSLT, XHTML, CSS and Javascript (for AJAX functionality). XQuery server pages can be executed from the filesystem or stored in the database.[2]

Released under GNU LGPL license, eXist is the most mature open-source XML native database written in Java. It acts similar to SQL-based tool, phpMyAdmin, as database administration tool. By installing eXist on your local computer, you will have almost entirely official eXist website (Figure 2). You have full access to all resources behind it, and for administration purpose you need to manage your user’s privilege and role. It also can used as a Java library to embed a database server in your own Java application, run it as a standalone database server as you would run a SQL database server, or run it embedded in a web server and get the features of both standalone database and a web interface to access the database[4].

Figure 2. eXist on your local computer

4. Implementation: Building Support Center Site

In this section, we will describe our implementation using eXist in building simple site in XML environment. As shown in figure 3, we will explain each step in separate sub-section. Site design and coding steps will combined in one part, and evaluation will be described more in next section.

4.1. Source analysis

As a case study, we decided to take some information from openSUSE Linux official site (http://en.opensuse.org)
4.2. Wrapper Generation

Because the existing data regularly showed as HTML, so we need one additional process to retrieve and transform them to XML format. By using manual coding approach in Python, we have developed our specific wrapper. The wrapper that we have developed is based on structure of HTML format of openSUSE official site known in 4.1.

4.3. Information Extraction

Having completed feature of XML-related technologies, eXist has also an ability to access website online. By using XQuery and XPath, we can directly retrieve all information without need to save them on local computer. Unfortunately, eXist expects very strict rules regarding to the document that will be processed. We have tried to access several websites directly using eXist, but none of them can be easily accessed. All of our test sites gave some error messages about the well-formedness of the document.

By using our wrapper, we have extract some pages in the openSUSE official site and turn them in our expected XML format. We also add some tags (distro, source, and date_updated) for documentation purposes.

For example, in the second entry we have tagged form:

```xml
<h2>What are the goals of the openSUSE project?</h2>
<p>The openSUSE project has three main goals:</p>
<ul>
<li>Make openSUSE the easiest Linux for anyone to obtain and the most widely used Linux distribution</li>
<li>Leverage open source collaboration to make openSUSE the world’s most usable Linux distribution and desktop environment for new and experienced Linux users</li>
<li>Dramatically simplify and open the development and packaging processes to make openSUSE the platform of choice for Linux developers and software vendors</li>
</ul>
```

After we feed it to wrapper, then we have:

```xml
<title>What are the goals of the openSUSE project?</title>
<description>
The openSUSE project has three main goals:
<li>Make openSUSE the easiest Linux for anyone to obtain and the most widely used Linux distribution</li>
<li>Leverage open source collaboration to make openSUSE the world’s most usable Linux distribution and desktop environment for new and experienced Linux users</li>
<li>Dramatically simplify and open the development and packaging processes to make openSUSE the platform of choice for Linux developers and software vendors</li>
</description>
<distro>openSUSE</distro>
<source>http://en.opensuse.org/Frequently_Asked_Questions</source>
<date_updated>2008-8-18</date_updated>
```

Because of online access, time to get it complete is vary depend on internet connection. Alternatively, we also save the target pages to our local computer and access it offline. There are no result differences, since HTML as like as ordinary text file in nature.
4.4. Site building

After we got XML document in certain format, we will begin to build our site in eXist platform. This site is designed to give solution for user based on keyword they have entered, especially in openSUSE linux. Behavior of this site is similar to google in the way of navigating the search result.

eXist role in this section is similar to another web server, in common with PHP/MySQL-based web server (XAMPP, LAMP, etc.). Database file (mainly in XML), collected in one directory which can be called by using WebDAV in web-dav://localhost:8080/exist/webdav/db. All scripts are placed under directory [exist_home]/eXist/webapp. By default, eXist only recognize .xql or .xq as executable script format but you can define any format by modifying web.xml configuration file.

In this paper, we have built some scripts to form our Support Center Site. At the first time, front page of the site will be shown to get keyword from user (Figure 5). The XQuery snippet to retrieve all data based on keyword is:

```xquery
let $keyword:=request:get-parameter("keyword","")
return(
  for $i in doc("/db/maspras/wosoc.xml")/linux/record
  where $i[text:match-any(*, $keyword)]
  return(<h3><a href="execute.xql?title=$i/title">data($i/title)</a></h3>,
  <p data($i/description/line)</p>)
)
```

Figure 5. Front Page of Our Site

If users are interest in one entry, they can click it and page will be redirected to another page with parameter passing (Figure 6). The XQuery snippet to get more explanation of one entry is:

```xquery
let $keyword:=request:get-parameter("title",())
return(
  for $i in doc("/db/maspras/wosoc.xml")/linux/record
  where $i/title[contains(.,$keyword)]
  return($i)
)
```

Figure 6. Page Showing Selected Entry

And also we need XSLT to transform and style each tag of our XML data to (in this case) HTML format. eXist provides one built-in function to handle transformation using XSLT: transform() defined in namespace http://exist-db.org/xquery/transform. With this function, we only need to pass our XML structure and XSLT stylesheet and let eXist do the transformation.

5 Result and Discussion

Using eXist we have done to build simple support center site. The main problem in our implementation is in data preparation stage. In the previous section, with wrapper tool we need one additional process to prepare XML data from HTM format. It does not make use the ability of eXist to directly access HTML document, since in fact many sites are not follow the HTML rules especially in well-formedness. Because of this situation, to feed XML document that is extracted from HTML format, we need to manual code the wrapper. The main drawbacks were that the code was quite long and that assumptions concerning the structure of the page to be parsed were spread all over the code[3]. The extracted XML document is still need to be checked to its consistency, because there are many entries that not follow our specified format in section 4.3. If we do not do this step, so eXist will fail to access the document.

Since eXist is declarative language, we need to declare all of namespaces that we will use. In general, we have
no problem in using eXist because it offers a lot of built-in function in processing XML document and handling HTTP-related function. Because it built in full XML environment, we can easily utilize some XML-related technologies. In our implementation, we have use XQuery as core of the site to query and collect the structure of the result. XPath is used to navigate XQuery based on keyword and underlying XML document structure and because the XQuery result is in XML format, we make use XSLT to transform them to HTML format.

From this experiment, we also found one opportunity about the development of integrated support system. By collecting several sites as data source, then transform them in XML and locate the transformed document in one database we have one integrated system to support anyone who has trouble in a specific area.

6. Conclusions

XML is promising as the next database format since it offers interoperability and flexibility to exchange independently. The emerging of XML as database format is supported by XML-related technology, like XQuery, XPath, and XSLT. One of the most important factors that also support the increasing use of XML is that it can represent data structures which are difficult or inefficient to represent as relational data.

We have using eXist as our platform to develop XML-based site and has prove that eXist is a mature native XML database. It also provides a lot of functionality that support development of XML-based environment. As XML become broadly used to be database format, eXist has also considered to become a choosen XML database management tool.

References