Gn-Dtd: Innovative Way for Normalizing XML Document

Ms Jagruti Wankhade 1* Prof. Vijay Gulhane 2
1. Sipna’s college of Engg and Tech., S.G.B. Amravati University, Amravati (MS) India
2. Sipna’s college of Engg and Tech., S.G.B. Amravati University, Amravati (MS) India
*Jagruti_wankhade22@rediffmail.com, V_gulhane@rediffmail.com

Abstract-
As XML becomes widely used, dealing with redundancies in XML data has become an increasingly important issue. Redundantly stored information can lead not just to a higher data storage cost, but also to increased costs for data transfer and data manipulation, such data redundancies can lead to potential update anomalies. One way to avoid data redundancies is to employ good schema design based on known functional dependencies. This paper presents a graphical approach to model XML documents based on a Data Type Documentation called Graphical Notations-Data Type Documentation (GN-DTD). GN-DTD allows us to capture syntax and semantic of XML documents in a simple way but precise. Using various notations, the important features of XML documents such as elements, attributes, relationship, hierarchical structure, cardinality, sequence and disjunction between elements or attribute are visualized clearly at the schema level.

Keywords- XML Model, GN-DTD design, Normalization XML schema, Transformation Rules

1. INTRODUCTION

With the wide exploitation of the web and the accessibility of a huge amount of electronic data, XML (extensible Mark-up Language) has been used as a standard means of information representation and exchange over the web. Additionally, XML is currently used for many different types of applications which can be classified into two main categories [5,6]. The first application is called document centric XML and the other is called data centric XML. The document centric XML is used as a mark-up language for semi-structured text documents with mixed-content elements and comments. The data centric XML consists of regular structure data for automated processing and there are little or no element with mixed content, comments, and processing instruction. The current XML data models however do not pay sufficient attention to the Problem of representing the structure of XML documents. We believe, in order to present more sophisticated forms of XML documents structure, the schema such as DTD or XML schema must taken into account since it is used to define and validate XML documents structure. In our work, we consider DTD, as it has been widely well accepted and expressive enough for a large variety
of applications. Furthermore DTD is an early standard for XML, and many legacy XML documents structures are defined by DTDs.

In this paper, we proposed a graphical notation of DTD called GN-DTD to overcome the above limitations. The GN- DTD helps to arrange the content of XML documents in order to give a better understanding of DTD structures, to improve an XML design and normalization process as well. GN-DTD has richer syntax and structure which incorporate of attribute identity, simple data type, complex data type and relationship types between the elements. Furthermore, the semantic constraints that are important in XML documents are defined clearly and precisely to express the semantic expressiveness.

2. RELATED WORK

Major current XML data models use directed edge labelled graphs to represent XML documents and their Schemas. These models consist of nodes and directed edges which respectively represent XML element in the document and relationship among the element. These existing XML model can be categorised into: XML model to represent instance of XML document, XML model represent XML schema and XML model for representing both XML document and XML schema. Examples are DOM(document object model), OEM(object exchange model)[7], S3-GRAPH[2] and many more.

As a summary, data models such as OEM, DOM, DataGuide have been designed for the purpose of information or schema integration. The focus of these data models is on modelling the nested structure of semi structured data but not modelling the constraint that hold in the data. In constrast, data model such as S3-Graph, CM Hyper graph, EER, XML Trees and ORA-SS have been defined specifically for data management. Amongst these models, the notation of ORA-SS, semantic network model and EER notations are best to be adopted and applied in GN-DTD.

3. XML MODEL DESIGN

Consider the DTD in Fig. 1. The first line of DTD in Fig. 1 shows that department is the root of the DTD. While second line shows that department consists of sub element course. The semantic relationship between department and course is indicated by the symbol *, represents that department can consists of zero or many course for each department. The third line of the DTD shows that each element course has sub element title and element taken_by. Symbol “,” between them indicated that they must occur in sequence. The fourth line indicates that element course has an attribute cno. The keyword ‘#REQUIRED’ represents that the attribute cno must appear in every course while “ID” indicates that the value of cno is unique within XML document. The fifth line of the DTD shows that the keyword “PCDATA” to despite that element title has no sub element and it is a leaf element and has a string value.

```xml
<!DOCTYPE department[
<!ELEMENT department(course*)>
<!ELEMENT course(title,taken_by)>```
ITS related XML document confirms to dtd is as follows

```xml
<!DOCTYPE courses [ 
  <courses>
    <course cn0 = “csc101”>
      <title> XML database </title>
      <taken_by>
        <student sno = “112344”>
          <firstname> zurinahni</firstname>
          <lastname> zainol </lastname>
          <teacher>
            <teacher tno = “123”>
              <tname>Bing</tname>
            </teacher>
          </teacher>
        </student>
        <student sno = “112345”>
          <firstname> Azli </firstname>
          <teacher>
            <teacher tno = “123”>
              <tname>Bing</tname>
            </teacher>
          </teacher>
        </student>
      </taken_by>
    </course>
    <course cn0 = “csc102”>
      <title> Database Design </title>
      <taken_by>
        <student sno = “112344”>
          <firstname> zurinahni</firstname>
          <lastname>zainol</lastname>
          <teacher>
        </student>
      </taken_by>
    </course>
  ]>
</courses>
```

Fig1: DTD  STRUCTURE DESIGN
Any XML document that satisfies and conforms to this DTD is likely to contain data redundancies which may lead to update anomalies. For example, as shown in Figure 2, the lecturer named Bing who teaches the same course number (cno) csc101 is stored twice, which will lead to the updation anomalies. To avoid such problems, a set of rules should be provided when designing a DTD for XML documents.

4. TRANSFORMATION OF DTD INTO GN-DTD

GN-DTD emphasizes the representation of semantic constraints between the complex elements, simple elements and attributes clearly. GN-DTD represents the structure and the semantic constraints of the XML document in a schema level. GN-DTD has following basic components:

- A set of complex element node representing the element that have subelement
- A set of simple element nodes representing simple element that have no subelement
- A set of attributes nodes representing the attributes defines in ATTLIST.
- A semantic relationship between two nodes.
- A root node

Consider following DTD

```xml
<!DOCTYPE department[
  <!ELEMENT department(course*)>
  <!ELEMENT course(title, student*)>
  <!ATTLIST course cno ID #REQUIRED>
  <!ELEMENT title (#PCDATA)>
  <!ELEMENT student(fname|lname?, lecturer)>  
  <!ATTLIST student Sno ID #REQUIRED>
  <!ELEMENT fname(#PCDATA)>
  <!ELEMENT lname(#PCDATA)>
  <!ELEMENT lecturer(tname)>  
  <!ATTLIST lecturer tno ID #REQUIRED>
  <!ELEMENT tname (#PCDATA)>
]>
```
Following is the list of some notations used to represent GN-DTD:

<table>
<thead>
<tr>
<th>Notation</th>
<th>Meaning</th>
<th>Notation</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;name...&gt;</code></td>
<td>Complex element</td>
<td><code>(name, n, n)</code></td>
<td>Part-of link simple element, single value, CDATA</td>
</tr>
<tr>
<td><code>&lt;name...&gt;</code></td>
<td>Mandatory simple element, single value, CDATA</td>
<td><code>(name, n, 1)</code></td>
<td>Part-of link attribute (Complex element and attribute)</td>
</tr>
<tr>
<td><code>&lt;name...&gt;</code></td>
<td>Required simple element, multi value, CDATA</td>
<td><code>(name, n, m, n)</code></td>
<td>0-1 ary one-to-many inheritance relationship</td>
</tr>
<tr>
<td><code>&lt;name...&gt;</code></td>
<td>Optional simple element, single value, CDATA</td>
<td><code>(name, n, m, *)</code></td>
<td>0-1 ary many-to-many inheritance relationship</td>
</tr>
<tr>
<td><code>&lt;name...&gt;</code></td>
<td>Optional simple element, multi value, CDATA</td>
<td><code>(name, n, m, *)</code></td>
<td>0-1 ary many-to-one inheritance relationship</td>
</tr>
<tr>
<td>Composites Attribute</td>
<td></td>
<td></td>
<td>Sequence between set of relationship</td>
</tr>
<tr>
<td>Reference Attribute</td>
<td></td>
<td></td>
<td>Disjunction between set of relationship</td>
</tr>
<tr>
<td>Required Attribute</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5. Constraint Between Set Of Relationship

5.1 Sequence Between Set Of Child Element Nodes

Normally each complex element node consist a single attribute node or multi attribute node. We emphasize in our notation those node must be located first in the sequence before include other simple or complex elements node. To illustrate this, we draw a directed curved up arrow and labeled with {sequence} across all the set of relationship involved. Consider the following segment of DTD and its GN-DTD where attribute Sno is located at first position in the sequence of child elements.

```
<!ELEMENT student (fname,lname,grade)>
```
<!ATTLIST student Sno ID #REQUIRED>
<!ELEMENT fname(#PCDATA)>
<!ELEMENT lname(#PCDATA)>
<!ELEMENT grade(#PCDATA)>

5.2. Sequence Between The Set Of Sub Element

We have a set of sub elements that are in an exclusive “OR” [XOR] relationship to represent notation “|” in DTD. For example, for the complex element node student, only one of its sub elements which are fname or lname, to be appeared as its sub elements in the XML document. To illustrate this, we draw a line and labeled with [XOR] across all the set of relationship involved. Follows is a real example of application. <!ELEMENT chapter (page| citation| table)* > which is equivalent with <!ELEMENT chapter (page*| citation*| table*) >.

Following is the GN-DTD formation of DTD in fig 3
TO Better understand, consider the following DTD

```xml
<!DOCTYPE school[
<!ELEMENT school (course*|subject*)>
<!ELEMENT course(students*)> 
<!ATTLIST course cno ID #REQUIRED> 
<!ELEMENT subject(students*)> 
<!ATTLIST subject sno ID #REQUIRED> 
<!ELEMENT students (student*)> 
<!ELEMENT student ( tel?, address*,grade?)> 
<!ATTLIST student Sno ID #REQUIRED> 
NameCDATA #REQUIRED> 
<! ELEMENT tel (#PCDATA)> 
<! ELEMENT address (EMPTY)> 
<ATTLIST address Code (CDATA) #REQUIRED street (CDATA) #IMPLIED 
city(CDATA)#REQUIRED> 
<! ELEMENT grade (#PCDATA)> 
```
This is The main Diagramatical Representation of DTD on which we are going to apply the Normalization
Rules to delete all the redundancies,anomalies which makes the XML as a bad XML document.

6. NORMALIZATION RULES FOR GN-DTD

6.1 First Normal Form GN-DTD (1XNF GN-DTD)

The first normal form for GN-DTD is about finding unique identifier attributes for the complex
elements set, and checking that no node (complex element, simple element or attribute) actually
represents multiple values. To be in first normal form, each attribute, complex element or simple
element is not NULL and has a single label. More importantly, the primary key (unique identifier) for the
complex element must be defined.

a) Only one value for each simple element node or attribute node of GN-DTD can be stored. If there is
more than one value, we must add some new element nodes or attribute nodes to store them.
b) The root element of a GN-DTD model should be located at level 0 and the cardinality of the root
element node must be one.
c) Each set of complex element node in the
   GN-DTD has at least one key attribute node.

1.6.2 Second normal form (2XNF GN_DTD)

Some nodes need to be restructured. However they can then still be in a single GN-DTD. This is
possible in XML because XML supports hierarchies in a single document, while relational databases do
not support hierarchies in a single row. This is different from the relational second normal form (2NF), which requires one-to-many relationships to be in separate tables. The GN-DTD is in second normal form if and only if:

a) GN-DTD is in 1XNF.

b) There is no nested binary inheritance relationship or ternary inheritance relationship under many-to-many or one-to-many inheritance relationships with the following condition: For each nested set of complex element $\text{CE}_{l+1}$ of $\text{CE}_l$, and any key attribute (ATT) of $\text{CE}_l$, the key attribute and simple element of $\text{CE}_{l+1}$ is not partial dependent on ATT of complex element $\text{CE}_l$.

1.6.3 Third normal form (3XNF GN_DTD)

In the third normal form of the GN-DTD, making changes to one unique complex element node set would not affect the integrity of another complex element node sets. If needed, a complex element node set would be divided into two separate complex element node set. GN-DTD is in third normal form if and only if:

a) GN-DTD is in 2XNF.

b) There exists no nested inheritance relationship type of $n$-ary many-to-one or many-to-many under a one-to-many inheritance relationship set in GN-DTD and the following conditions are satisfied:

(i) For each nested set of complex elements $\text{CE}_{b,l+1}$ of set of complex element $\text{CE}_{a,l}$, any key attribute and simple element of $\text{CE}_{b,l+1}$ is not transitively dependent on ATT of complex element $\text{CE}_{a,l}$.

(ii) Any key attribute node of any complex element node located in a different level are disjoint (ATT$\text{CE}_l$ ∩ ATT$\text{CE}_{l+1}$ ∩ ATT$\text{CE}_n$ = 0).

1.6.4 Normal form GN-DTD(NF GN-DTD)

GN-DTD is in Normal Form if and only if:

a) GN-DTD is in 3NF.

b) There are no global dependencies between attribute and simple element of complex element nodes under nested one-to-many or many-to-many inheritance relationship.

7. TRANSFORMATION FROM GN-DTD TO DTD

After removing all the types of redundancies GN-DTD can be transform back to DTD structures. Following is the set of some transformation rules used to come back to the original DTD.

Step 1 Level 0, a root node is represented By <!DOCTYPE root node name [element type definition] >

Step 2 Level 1, identity the sub tree of GN-DTD check the number of nodes, type of nodes and relationship type

Step 3 If there is no more than one node at level 1 and nodes are hierarchical then generate

$<$ELEMENT root node name (N1) $>$ Where N1 is the list of sub elements/child nodes

3.1 Check the relationship set between parent Nodes and child nodes.
3.1.1 If \{XOR\} means the relationship between node is a disjunction and will be represented using symbol ‘|’ Else

3.1.2 If \{sequence\} means the relationship is sequence and will be represented using symbol ‘,’

3.2 Check the semantic constraint between parent nodes and child nodes in each of relationship set and map to following operator:

3.2.1 if \[[0..N]\] map to operator *,

3.2.2 if \[[1..N]\] map to operator +

3.2.3 if \[[0..1]\] map to operator ?

Step 4 If the list of sub elements \(Ni\) is not empty, using depth first traversal, for each node in list sub element \(Ni\)

4.1 repeat step 3.1 and 3.2

4.2 generate \(< ! ELEMENT Ni (sub element Nj)>\)

4.3 for each complex element \(Ni\), find an attribute node and generate

\(<!\text{ATTLIST } Ni \text{ attribute name attribute type}>\)

4. For sub element \(Nj\)

4.4.1 If \(Nj\) is a simple element has part of link with \(Ni\) then generate

\(<!\text{ELEMENT simple element name CDATA}>\)

(Repeat for all simple element nodes)

4.4.2 If \(Nj\) is a complex element node has inheritance link with \(Ni\) Repeat step 4

4.4.3 If \(Nj\) is a complex element node has part of link then generate

\(<\!ELEMENT Nj (EMPTY) >\)

Step 5 Go to next sub tree GN-DTD and repeat step 4

1.7 CONCLUSION

We have proposed a method for designing a “good” XML document in two steps: first, we building a conceptual model by means of GN-DTD at the schema level and second, using normalization theory where functional dependencies are refined among its simple elements and attributes. The GN-DTD can be further normalised either to 1XNF, 2XNF, 3XNF or XNF using the proposed normalization algorithm. In the proposed methodology, a GN-DTD is used as input and the normalization rules are applied during the normalization process. We also explain the process for transforming GN-DTD into DTD.

1.8 REFERENCES


Author Biography:

1] Miss. Jagruti Wankhade
   sipna’s college of Engg and Tech, Amravati
   S.G.B. Amravati University, (MS), India

2] Prof. Vijay Gulhane
   B.E.(CMPS), M.E.(CMPS), PhD (pursuing)
   S.G.B. Amravati University, (MS), India
   Working as a (A.P.) in sipna’s college of Engg and Tech, Amravati
This academic article was published by The International Institute for Science, Technology and Education (IISTE). The IISTE is a pioneer in the Open Access Publishing service based in the U.S. and Europe. The aim of the institute is Accelerating Global Knowledge Sharing.

More information about the publisher can be found in the IISTE’s homepage: http://www.iiste.org

The IISTE is currently hosting more than 30 peer-reviewed academic journals and collaborating with academic institutions around the world. **Prospective authors of IISTE journals can find the submission instruction on the following page:** http://www.iiste.org/Journals/

The IISTE editorial team promises to the review and publish all the qualified submissions in a fast manner. All the journals articles are available online to the readers all over the world without financial, legal, or technical barriers other than those inseparable from gaining access to the internet itself. Printed version of the journals is also available upon request of readers and authors.

**IISTE Knowledge Sharing Partners**

EBSCO, Index Copernicus, Ulrich's Periodicals Directory, JournalTOCS, PKP Open Archives Harvester, Bielefeld Academic Search Engine, Elektronische Zeitschriftenbibliothek EZB, Open J-Gate, OCLC WorldCat, Universe Digital Library, NewJour, Google Scholar