XLM 文档转换为关系数据的一种方法

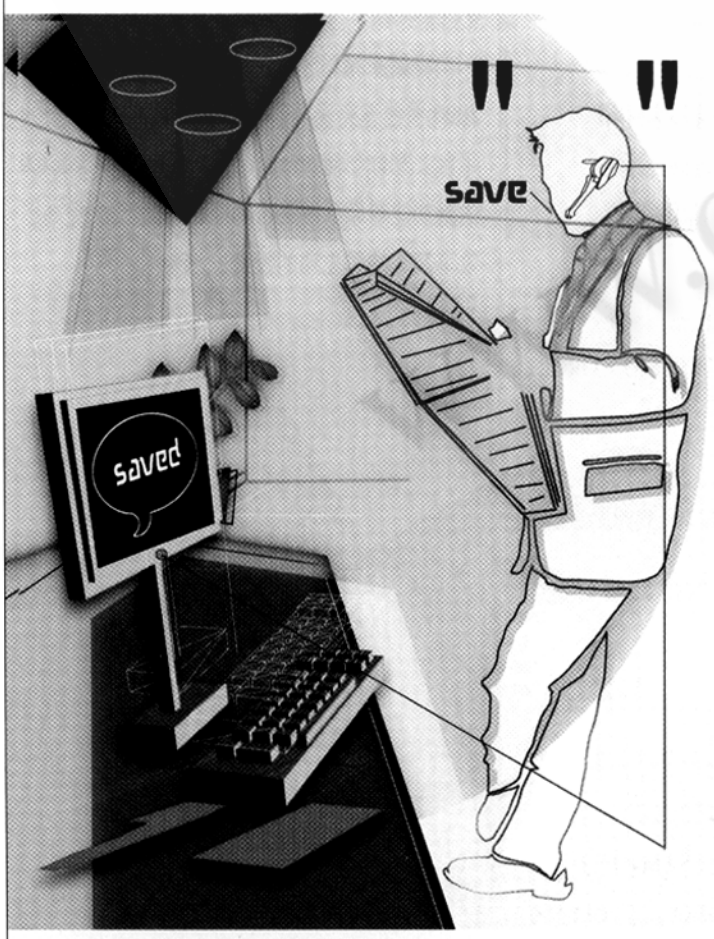
1 XML 简介

XML 是专为 Web 应用而设计的 SGML（Standard Generalized Markup Language）的一个优化子集。XML 是一种元标记语言，使用者可按需创建新的标记，XML的可扩展性就在于此。这些标记通过XML DTD（Document Type Definition）来加以定义。它定义了文档所需的标记，比如可在文档里使用的元素类型，这些元素之间的可能的联系，还可声明元素的属性，属性是元素的"元数据"，而不是元素包含的内容。XML 文档可以在它的文档类型声明里声明该文档遵循某个XML DTD。根据对XML的理解，其具有下列特性：

(1) 可扩展性 XML 是设计标记语言的元语言，而 HTML 是只有一个固定标记集的特定的标记语言。

(2) 灵活性 XML 提供了一种结构化的数据表示方式，使得用户界面分离于结构化数据。

(3) 自描述性 XML 文档通常包含一个文档类型声明，不仅人能读懂 XML 文档，而且计算机也能处理。XML 文档中的数据可以被任何能够对 XML数据进行解析的应用所提取、分析和处理，并以所需格式显示，XML 表示数据的方式真正做到了独立于应用系统，并且这些数据能重用。



2 XML 文档到关系数据的转换

实现XML文档到关系数据的转换，首先要实现从XML的数据模式到关系数据模式的转换，即将XML数据模式到关系模式的转换方法描述成文本格式的转换标准，当数据交换时依据该转换标准进行转换。

2.1 DTD 与 ER-Model

由于目前使用的XML的数据模式主要是DTD，它包含了标注的声明，是一类XML文档的文法（Grammar）；是一种声明和定义相关文档实例的机制。具体地说，DTD定义了文档实例的每一个标注及其相互关系，从这一点来看很象ER图中的实体和关系的关系。

下面，首先给出一个出版信息的DTD描述。

```
<!ELEMENT book (booktitle, author)>
<!ELEMENT article (title, author *, contactauthor)>
<!ELEMENT contactauthor EMPTY>
    <!ATTLIST contactauthor authorID IDREF IMPLIED>
<!ELEMENT monograph (title , author , editor)>
<!ELEMENT editor (monograph*, name)>
<!ELEMENT author (name, address)>
    <!ATTLIST author id ID #REQUIRED>
<!ELEMENT name (firstname?, lastname)>
<!ELEMENT firstname (#PCDATA)>
<!ELEMENT lastname (#PCDATA)>
<!ELEMENT address ANY>
```

由于ER-Model是以图的形式存在的，为了更好地分析和对应DTD和ER-Model，先引入DTD图。DTD图可以用来表示DTD的结构。DTD的结点是DTD中的元素（Element）、属性（Attribute）和操作符。每一个元素在DTD图中能且只能出现一次，而属性和操作符则可出现多次，这依据于它在DTD中出现的次数。DTD的边来源于DTD文档中的每一行定义，一个DTD关系对应一个DTD图中的边。例如，前面的出版信息的DTD描述可以被转化为如图1的DTD图。其中的环表明该DTD中存在递归。

从图1可以看出DTD到DTD图的转换是很直接的。根据图1，可设计

摘要：本文介绍了一种将DTD转换成ER图，并用XML Application将ER图描述成转换标准，然后根据该转换标准将XML文档转换为关系模型的方法。

关键词：可扩展标注语言　文档类型定义　数据交换　关系数据库

张 哲 （浙江绍兴文理学院计算机系　312000）

出图2的ER模型。可以看出从DTD图到ER-Model主要是解决嵌套XML结构的实体平展化和几个算子的关系化的问题。例如上例中的DTD中author的所有子元素被平展到ER中author的属性。而book则并没有把author也平展进去，因为author实体是必须的。图1中的两个*算子分别被转换成Editor与Monograph的多对多关系及Article与Author的多对多关系。

2.2 根据转换标准将XML文档转换为关系模型

将XML文档转换到关系表的规则定义使用XML来描述，作为转换标准。

2.2.1 XML文档到关系模型的转换步骤

(1) 先将其DTD转换为DTD图；

(2) 再将DTD图转换为ER图；

(3) 然后使用XML Application来描述ER图，并将其作为转换标准；

(4) 当需要转换XML文档到关系表时，依据转换标准进行转换。

2.2.2 转换标准

转换标准的实质应当是一种描述ER图的XML Application。依旧使用出版信息的例子来讨论从XML文档到关系数据库的转换。前面已给出了DTD转换成DTD图，DTD图转换成ER图。下面首先给出描述ER图的XML Application的DTD定义，然后利用该XML Application对ER图中的实体及关系进行描述。

ERTransFormat的DTD定义如下：

```
<!DOCTYPE ERTransFormat  [
<!ELEMENT ERTransFormat (ENTITY*, RELATION*)>
<!ELEMENT ENTITY (STRUCTURE, GENERATION)>
    <!ATTLIST ENTITY ID ID #REQUIRED>
<!ELEMENT RELATION (STRUCTURE, GENERATION)>
    <!ATTLIST RELATION ID ID #REQUIRED>
<!ELEMENT STRUCTURE (FIELD*)>
<!ELEMENT FIELD (FOREIGN_KEY)>
    <!ATTLIST FIELD ID ID #REQUIRED>
    <!ATTLIST FIELD type NOTATIONS (CHARINUMBERIDATEITEXT)>
    <!ATTLIST FIELD length CDATA #REQUIRED>
    <!ATTLIST FIELD primary_key NOTATIONS (yesIno)>
    <!ATTLIST FIELD autogenerate NOTATIONS (yesIno)>
<!ELEMENT FOREIGN_KEY EMPTY>
```

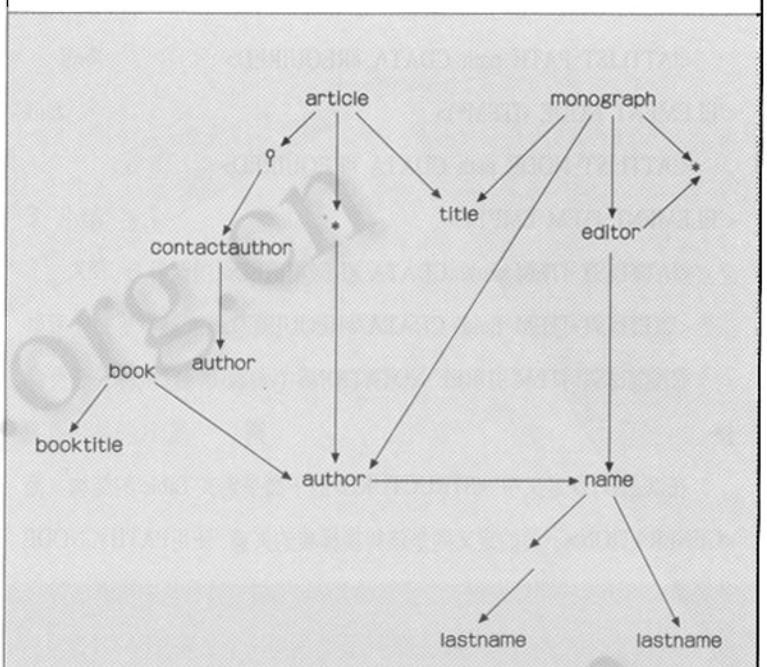


图1 An Example of DTD Graph

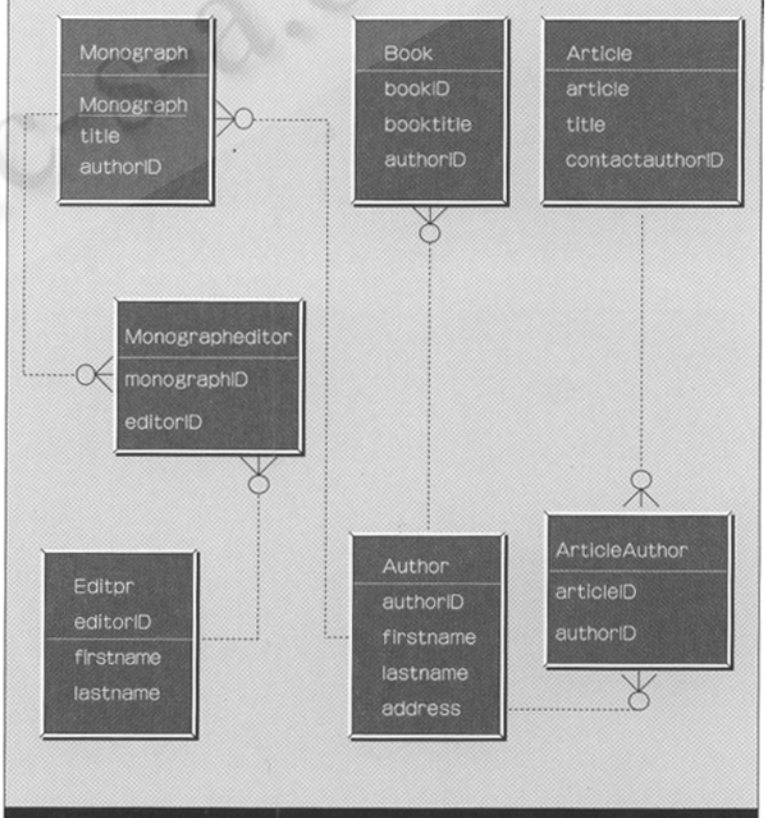


图2 ER-Model for DTD

```
    <!ATTLIST FOREIGN_KEY entity CDATA #REQUIRED>
    <!ATTLIST FOREIGN_KEY field CDATA #REQUIRED>
<!ELEMENT GENERATION (PATH+)>
<!ELEMENT PATH (NODE*)>
  <!ATTLIST PATH path CDATA #REQUIRED>
<!ELEMENT NODE (ITEM*)>
    <!ATTLIST NODE path CDATA #REQUIRED>
<!ELEMENT ITEM EMPTY>
    <!ATTLIST ITEM path CDATA #REQUIRED>
    <!ATTLIST ITEM field CDATA #REQUIRED>
    <!ATTLIST ITEM IDREF NOTATIONS (yes|no)>
]>
```

在以上DTD定义中，<STRUCTURE>的下面是定义Table的结构，而<GENERATION>下面的定义则是该转换标准的关键，使用PATH和NODE的嵌套来挑选出将要转换到一个元组的XML子树，然后从中挑选出N个属性或者属性对应的引用来生成元组。两个有同一个父亲的PATH是两种生成规则，满足其一即可。

其中，path语法如下：

path ::= $(path)_{Ident}$ | path.path | node | *

node ::= NULL | tag | tag*attribute

*表示任意路径，即通配符，#来表示后面的元素是前面元素的一个属性，.是路径的元素分割点。

而PATH、NODE、ITEM和IDREF的详细使用如下：

PATH：生成元组的第一步，元素定位，PATH中的属性path帮助系统定位到合适的转换点（后缀匹配）。

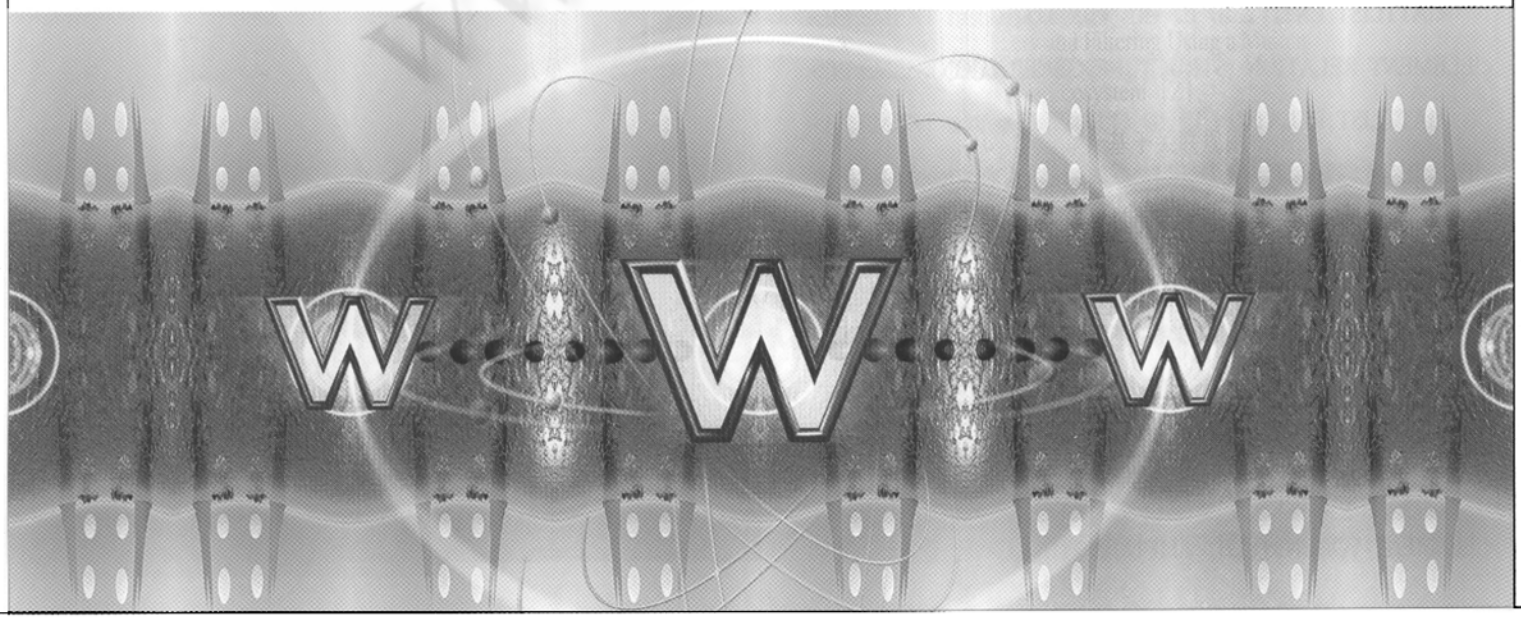
NODE：定义一组字段的转换，它们一般都存在于PATH中的path的某个结点下。因此NODE中的path一定是PATH中path的无二义子路径。

ITEM：详细定义某一字段的转换，使用的是相对NODE中path的路径描述来定位待转换的元素（Element）或属性（Attribute）。

IDREF：表明这样一个ITEM并不关心结点下的内容，关心的是该结点下生成的其他关系中元组的引用，其受FOREIGN_KEY描述的限制。

下面以出版信息ER图中的Author实体为例给出其XML Application描述

```
<ERTransFormat>
<!-ER-Model Transformation Format //-->
  <ENTITY ID= "Author">
    <STRUCTURE>
      <FIELD ID= "authorID" type= "CHAR" length= "20"
primary_key= "yes" />
      <FIELD ID= "firstname" type= "CHAR" length= "30" />
      <FIELD ID= "lastname" type= "CHAR" length= "30" />
      <FIELD ID= "address" type= "CHAR" length= "60" />
  </STRUCTURE>
  <GENERATION>
    <PATH path= "author">
      <NODE path= "author">
        <ITEM path= "#id" field= "authored">
        <ITEM path= ".name.firstname" field= "firstname">
        <ITEM path= ".name.lastname" field= "lastname">
        <ITEM path= ".address" field= "address">
      </NODE>
```



```
    </PATH>
  </GENERATION>
</ENTITY>
```

以上实体描述对应ER-Model中的Author，<STRUCTURE>子文档描述的是关系，即目标数据库中的表结构，而<GENERATION>子文档则是描述怎样从一个XML文档中将数据抽取到关系库的方法。<PATH path="author">表明当在遍历XML树的时候若当前路径的后缀为author则在关系Author中新增一个元组，并记该结点为[A]，<NODE path="author">表明以下是结点匹配，path中的author对应的是PATH的属性path中author所对应的结点[A]，对于NODE下一层的ITEM，属性path是以NODE的path为起始点的相对路径，如<ITEM path=".name.firstname" field="firstname">表明是将[A].name.firstname的内容转换到字段firstname中去。而<ITEM path="#id" field="authorID">则表明将[A]#id的内容转换到字段authorID中去，#是path中的属性算子，表明id是[A]的一个属性。

在该出版信息中，其他实体及关系的XML Application描述与以上的实体Author类似，就不再给出了。

最后，给出将XML文档根据转换标准转换成关系模型的算法概述：

算法1：关系模型生成算法

假设待转换的XML文档已经被分析转换成DOM对象树，根为Xroot。

```
Call XML2Relation (XRoot , XRoot^.tagName)

Procedure XML2Relation (p , path)
Begin
  q := p^.firstchild;
  while ( q<>nil ) Do Begin
    XML2Relation (q, path+'.'+q^.tagName);
    q := q^.nextSibling;
End;
For ( e in Entity or Relation ) do
  If path的后缀与e的<GENERATION>下的某一个PATH（ph）匹配
Then Begin
    For （node in Node of ph）do begin
      根据PATH的path和NODE的子项path定位到待转换的XML文档的结点x；
      For （item in Item of node ）do begin
        将x..（ITEM#path)对应的元素转换为关系e中的字段(ITEM#field)
          若IDREF="yes"，则填为对应外键引用。
        End；
      End；
    End；
End；
```

3 小结

XML这种新的Web数据组织形式，在Internet和电子商务中的数据交换等方面变得越来越重要。然而，XML不会替代传统的关系数据库，而是会更充分利用它们的强大功能。本文提出了一种在XML和关系数据库之间实现转换的方法。

参考文献

1 W3C.The XML1.0 Recommendation [S] 2000.8 .Second Edition, http://www.w3.org/TR/REC。
2 Charles F.Goldfarb,Paul Prescod. The XML handbook [M]. Prentice Hall, 1998。
3 Steven Holzner. XML COMPLETE [M]. The McGraw-Hill Companies, Inc.1998。
4 Robin Cover. XMLMetadata Interchang(XMI).http://www.oasis-open.org/cover/xmi.html。
5 Barclay T. Blair, John Boyer, XFDL:Creating Electronic Commerce Transaction Records Using XML. International World Wide Web Conference.1999。
6 Ronald, XML and Database . http://www.rpbourret.com/xml/xml and Databases.htm .2001。
7 Dan Connolly The XML Revolution. http://www.xml.org.cn。
8 Ronald Bourret XML Database Productions http://www.rbourret.com/xml/XMLDatabaseProds.htm。
9 Ronald Bourret Mapping DTDs to Databases http://www.xml.com/pub/a/2001/05/09/dtdtodbs.html。
10 Ronald Bourret Mapping W3C Schemas to Object Schemas to Relational Schemas http://www.rpbouret.com/xml/SchemaMap.htm。