一种基于 XML 的 Web 页面定义语言*

李效东

(中国科学院 软件研究所,北京 100080)
E-mail: xiaodong_li@hotmail.com
http://www.ios.ac.cn

摘要: 数据密集(data-intensive)型 Web 站点是指那些将大量的异构数据源的数据进行集成以后所生成的 Web 站点.此类站点的建设可以划分为 3 项主要任务:数据的访问与集成、站点结构的构造(指定各页面所含内容以及各页之间的链接)和页面的 HTML 表示.以 XML 以及来自 W3C 的相关规范为基础,构建了一种 Web 页面定义语言 WPDL(Web page definition language).给出了 WPDL 的一个简化的 EBNF 表示,并举例说明了 WPDL 的关键特性.并且说明,使用这种说明式的查询语言(declarative query language),可以将数据访问、站点结构生成与页面表示彼此分离开来,使站点的重构(restructuring)、重用(reusability)和完整性约束的实施(integrity constraint enforcement)成为可能.

关 键 词: 说明式查询语言;数据密集型站点;页面树;WPDL;XML;XSL
中图法分类号: TP393 文献标识码: A

数据密集型站点的建设是软件工程研究所面临的重要而紧迫的课题之一.数据密集型站点的建设一般可以划分为 3 项主要任务:一是对异构数据源数据的访问和集成;二是进行站点结构的构造,即指定各页面所含内容以及各页面之间的内外部链接;三是生成页面的 HTML 表示或 XML(extensible markup language)加样式表语言(如 XSL[1](extensible style sheet language))或 CSS[2](cascading style sheets)的表示.目前,Web 站点对处于其下层数据源的访问集成主要通过 CGI(common gateway interface)技术,其主要特点是数据源数据的获取和 Web 页面的定义与生成由一些特定(ad hoc)的程序来完成,3 项任务之间纠缠在一起非常不利于站点的重构、扩展和优化.具体来说,将数据密集型站点建设的 3 个阶段独立开来,有一些明显的好处.第 1,有利于生成站点的不同版本.例如,对不同种类的用户,虽然数据源相同,但却希望给他们呈现不同的页面内容和形式.如果利用 CGI 技术,我们可能不得不针对不同的用户编写不同的程序,这就没有任何重用性可言.第 2,将 3 个任务分离,我们必须设计一种 Web 站点的中间逻辑表示方式,在这种中间逻辑表示的基础上,我们就可以从较高的层次上来处理 Web 站点问题.例如,可以开发算法用于站点的完整性约束(integrity constraints)检验[3],甚至是自适应 Web 站点(adaptive Web-site)的构建[4]等.第 3,便于对站点的自动更新.我们只要更新数据源,然后周期性地执行查询语句即可.第 4,便于将新的数据源集成到 Web 站点系统中.对于结构化数据,如关系数据库系统,只要它支持标准的数据库查询语言,如 SQL(structured query language),就很容易集成到 3 项任务彼此间独立的 Web 站点系统中.而对于非结构化数据或半结构化数据,只要通过打包程序(wrapper)生成一致的表示形式,如 XML 文档形式,同样可以很容易地集成到这样的 Web 站点系统之中.

所谓基于 XML 技术[5]有两方面的含义:首先是 WPDL 不但针对结构化数据而设计,而且面向半结构化数据[6],而本文所探讨的半结构化数据是 XML 文档数据;其次,WPDL 的执行结果仍然是 XML 文档,在把其转换成可浏览页面时,仍然需要借助与 XML 技术有关的标准.本文涉及的是 XSL.

我们是从数据库的角度,而不是从文档的角度来看待 XML 文件,也就是说,将 XML 文档本身看成是数据库,

* 收稿日期: 2000-03-09; 修改日期: 2000-07-05
作者简介: 李效东(1968－),男,辽宁黑山人,博士生,讲师,主要研究领域为数据库技术的 Web 应用.

而将其对应的 DTD(data type definition)看成是数据库模式(schema).在 DTD 上实施查询就相当于在数据库 schema 的基础上来实施查询.

1 数据模型——XML 树(XTree)

考虑到 XML 这种半结构化数据表示方式的特点,我们将文献[7]提出的针对半结构化数据的数据模型作一些必要的扩展,以适应表示 XML 文档.

首先,我们简单地介绍一下半结构化数据(semi-structured data).有关半结构化数据并没有一个严格和统一的定义.一般来说,所谓半结构化数据是指那些信息和描述信息的模式(schema)纠缠在一起的数据,有时称为具有自描述特征(self-describing)的数据.它们既不像原始数据(raw data)那样毫无结构可言,又不像标准数据库系统中那些严格类型化(strictly typed)的有分离模式(separate schema)来约束的数据.在半结构化数据中,结构是存在的,它们松散地约束着数据,结构必须从数据中抽取才能得到.由于传统数据库系统的局限性,近年来半结构化数据一直是一个非常流行的研究课题[8].XML 文档是典型的半结构化数据,对半结构化数据的研究为它的进一步发展奠定了坚实的理论基础.

本文之所以引进以下要描述的数据模型,有两方面的意义.首先,WPDL 语句中的查询部分(见本文第 2 节和第 3 节中详细描述)的查询语句必须建立在此数据模型的基础之上;其次,WPDL 语句中的构造部分的目的,就是要将查询结果组合成新的模型表示.而这个模型表示实际上是一个站点的说明性规范说明(declarative specification),亦即前面所说的中间逻辑表示方式.根据它,我们就可以进行站点的完整性约束检验[3],甚至是自适应 Web 站点的建立[4].

将半结构化数据表示成某种图状(graph-like)或树状(tree-like)结构,是研究界较普遍的认识[4,5].本文对文献[7]提出的数据模型和查询语言加以改造和扩展,以适应 XML 的数据表示.本文提出的半结构化数据模型实际上是一种"边加标签的带根有向图 (an edge-labeled rooted directed graph)".下面先给出它的形式化定义,然后在此基础上引申出 XML 数据模型 XTree 的定义.

定义 **1(**边加标签的带根有向图**)**. 一个带根、边加标签的有向图是一个四元组 $G=(V,E,r,\lambda)$.其中 V 是一个非空的结点集合;$E\subseteq V\times V$ 是图中边的集合;(V,E)代表一个有向多重图(directed multi-graph);$r\in V$ 表示根,并且满足:任意一个 V 中的节点,从 r 开始经过 E 中的边都能够到达;$\lambda:E\rightarrow Label$ 是一个函数,表示边集合 E 到标签集合 Label 的映射.

定义 **2(Xtree** 数据模型**)**. 论域 D 是包括所有字符串类型数据的集合,D 上的所有带根、边加标签有向图组成的集合用 T 来表示 $T\subseteq G$,T 中的元素 t 要么是一个原子数据,即 $t=d\in D$;要么 $t=d[t_1,...,t_n]$,其中 $d\in D,t_1,...,t_n\in T$,称 d 为标签(label),$t_1,...,t_n$ 为 t 的子树.

$t=d[t_1,...,t_n]$对应到 XML 文档数据,称 t 为 XML 树,简称 XTree,它有如下属性:

·树的每一个节点都有一个唯一的标识符(ID).这个标识符可以显式地以 XML 文档某一元素的 ID 属性来标识,也可以为其分配一个唯一的 ID 来标识;

·以 XML 的术语 t 为元素(element),非叶标签 d 为元素类型(element type),$t_1,...,t_n$ 为 t 的子元素(subelements);叶标签 d 为原子对象时,为文字值或空元素;

·当考虑$[t_1,...,t_n]$中各子树的顺序时,称为顺序的 XTree(ordered XTree)模型,当不考虑$[t_1,...,t_n]$中各子树的顺序时,称为无顺序的 XTree(unordered XTree)模型.

图 1 是一个 XML 文档实例,图 2 是该文档的 XTree 模型表示示意图.限于篇幅,此处没有给出该文档的 XML DTD.

路径表达式是 XML 查询的基本建筑单元.针对模型 Xtree,我们给出其形式化定义:

定义 **3(**路径表达式**)**.

(1) {}表示一棵空树;

(2) $\{l=>T\}$表示一棵树,它的根有一条标记为 l 的输出边,下面附着一棵子树 T;其中 T 可以表示叶节点,此时 T 等于某文字值或空元素.由于$\{l=>T\}$只表示单一的一棵树,可省略括号,简写为 $l=>T$;

(3) $\{l_1=>T_1,l_2=>T_2,...,l_n=>T_n\}$也表示一棵树,它有 n 个由同一节点发出的分别标记为 $l_1,l_2,...,l_n$ 的输出边,每个输出边分别附着子树 $T_1,T_2,...,T_n$;

(4) 无论是边标签 l,还是子树 T_i(包括叶节点),都可以用变量来表示;

(5) (2),(3)或者(2)与(3)的组合称为路径表达式.

```
⟨bib⟩
  ⟨book⟩
    ⟨title⟩A Great Book⟨/title⟩
      …
    ⟨publisher⟩
      ⟨publishername⟩Pt Hall⟨/publishername⟩
      ⟨address⟩No8 Tree Street⟨/address⟩
    ⟨/publisher⟩
  ⟨/book⟩
⟨book⟩
  ⟨title⟩A Good Start⟨/title⟩
    …
  ⟨publishername⟩Wrox⟨/publishername⟩
⟨/book⟩
⟨article⟩
  …
⟨/article⟩
…      ⟨/bib⟩
```

Fig.1 An example of XML document
图 1 一个 XML 数据文档实例

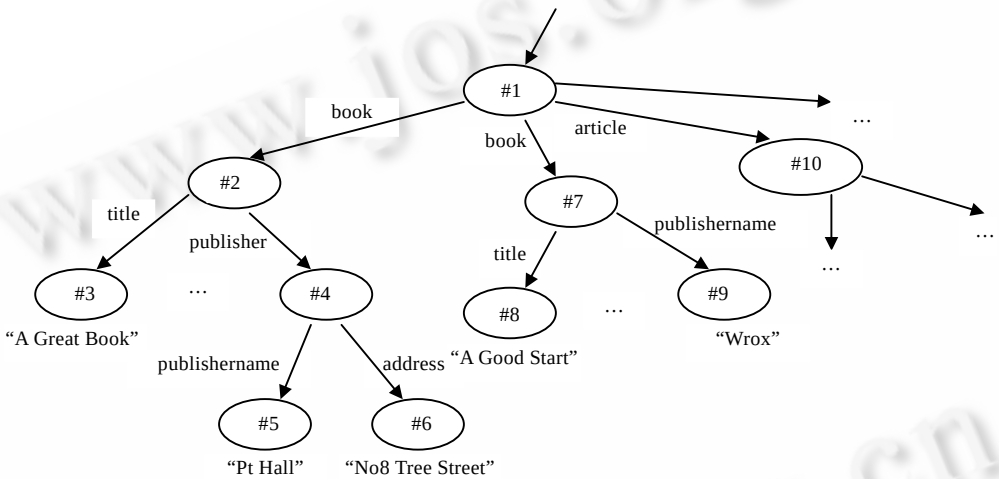

Fig.2 An example of Xtree modeling
图 2 一个 XML 文档 XTree 模型表示示意图

2 XML 查询语言 AnXQL

AnXQL 是作者设计的另一种 XML 文档查询语言[9](another XML query language).因为在 WPDL 的查询部分要用到它,所以本节先对其加以介绍.设计 XML 文档查询语言的目的,是从数据库的角度来看待 XML 文档,从文档中抽取出用户所感兴趣的信息.AnXQL 查询语言的设计试图突破关系型数据库中的一阶逻辑和第一范式的限制.一阶逻辑,即关系演算/代数,是关系型查询语言的出发点.然而在对待非关系数据结构时,比如复杂的面向对象数据,关系型查询语言就显得力不从心.关系系统中的第一范式要求不能有关系的嵌套,这样就很难用关系查询语言来对待复杂对象数据.人们期待查询语言具有更强的表达能力.

AnXQL 查询语句由两个重要的部分组成,分别称为构造部分(construction part)和抽取部分(extraction part).抽取部分由模式匹配(pattern matching)条件表达式和称为过滤器(filter)的谓词条件表达式组成.构造部分由 select 子句来表达,而抽取部分由 where 子句来表达.所谓模式匹配就是用路径表达式来定义查询条件,将路径表达式作为一个模式到原文档中去匹配,满足条件的 XML 文档片断被返回;而谓词表达式是指结果为布尔类型的条件表达式.对于 AnXQL 的详细描述,我们将另文给出,这里只举几个实例说明它的一些关键特性.

例如,针对与图 1 的 XML 文档,可能有查询请求 Q1 如下:

```
SELECT result=>{book=>{title=>$x,price=>$y,author=>$z}}
```

```
WHERE bib=>{book=>{title=>$x,price=>$y,author=>$z}}
AND $y<=50 IN "www.a.b.c/bib.xml";
```

其中,WHERE 子句由一个路径表达式和一个谓词表达式($y<=50)组成.如果进一步细分,查询 Q1 实际上由 3 部分组成:构造部分、匹配部分和过滤器部分.3 个部分之间,数据的传递可以视为元组(tuple)的传递,元组具有扁平和顺序无关的特性,这是 AnXQL 查询语言的一个重要特征."IN"子句用来指定查询的文档所在的 URI(unified resource identifier).为简单起见,对单一数据源查询的 AnXQL 语句省略"IN"子句.首先,路径表达式到原文档中去匹配,结果是满足匹配模式的值绑定到元组集合($x,$y,$z),然后元组集合被传递给过滤器,变量$y 小于等于 50 所对应的元组被保留,并被传递给构造部分,按 select 子句的路径表达式为每一组元组加标签生成新的 XML 模型表示.

3 Web 页面定义语言 WPDL

WPDL 面向的数据源主要包括两类:一类是被广泛使用的关系型数据库数据源,另一类是正在大量出现并使用的 XML 数据源.后一种数据源实际上是半结构化数据的一个特例.而且很容易对 WPDL 进行扩展,以集成其他种类的数据源,其中重要的方法之一是将半结构化数据源用特定的打包器转换成相应的 XML 文档.而对于类似的关系型数据库数据源的元组流数据,则可借助于成熟的技术,如 JDBC 和 ODBC.对于关系型数据源,我们使用 SQL 语言来实施查询;而对于 XML 数据源,则使用在本文前面所提出的 AnXQL 来实施查询.

WPDL 的语句块由两部分组成,一个称为查询部分,另一个称为构造部分.查询部分负责从各个异构数据源获取数据.查询部分的查询语句执行以后,返回一个元组集合(关系).构造部分将这个元组集合构造成一个 XTree 表示.每个 WPDL 语句构造的 XTree 通过内部链接构成站点图.

下面给出了用 EBNF 表示的 WPDL 的一个语法概要(对可以进一步展开的项没有向下展开).我们将举例说明对它如何使用.

```
WPDL::=
    CREATE PAGE(HomePage()|
        ⟨PageName⟩([URL(⟨Varible⟩|⟨Constant⟩)])
    AS
        ⟨PageTrees⟩[;⟨Assignments⟩]
    USING
        ⟨Variables⟩ IN ⟨AnXQL⟩|⟨Variables⟩ IN ⟨SQL⟩
```

我们从下向上依次介绍各部分的含义.首先,从 USING 子句开始,它是 WPDL 的查询部分.面向不同的数据源使用不同的查询语言,对于关系型数据库数据源使用 SQL,而对于 XML 文档数据源使用 AnXQL 查询语言.查询返回的结果构成 XML 文档的片断或称为 XTree 的子树.

AS 子句是 WPDL 的构造部分,它使用在本文第 1 节中介绍的语法来构造新的 XTree(见下面的例子),实际上是 XML 文档的一种表现形式.

CREATE PAGE 子句是 WPDL 语句的开始,类似于 SQL 语言中的视图定义语句 CREATE VIEW.其后用"|"符号隔开的,是两个可选的项目.其中 HomePage()是一个保留字,用以说明现在创建的是起始页(主页).是⟨PageName⟩代表用户为除了起始页以外的任意新创建的 XML 页面所取的内部名.主要用于在 XTree 中标识页面间的链接.URL()是一个 Skolem 函数,该函数的参数可以是由用户指定的常数,也可以是来自查询部分的变量.

下面引入一个例子来说明 WPDL 语言的使用情况.这个例子是一个虚拟的例子,不一定与实际情况相符,但足以说明问题.例子中假设构造一个主题为"计算机科学家和他们的书"的 Web 站点.构成新站点的数据来源于两个数据源,一个是存储计算机科学家人事信息的关系型数据库 COMPUTIST,另一个是收集计算机科学参考文献的站点(简写为 S1),它以 XML 的方式发布 Web 数据.图 3 是科学家数据库的其中两个基表的模式,图 4 是参考文献站点的其中某一类 XML 文档的 DTD.为了方便描述问题,假设一个科学家仅从属于某一固定的研究领域.

```
Scientist(firstname,lastname,specialty,position,address,email,photo)
ScientistInInstitution(firstname,lastname,institution,website)
```

Fig 3 The schema for relational database COMPUTIST
图 3 关系型数据库 COMPUTIST 的模式

```
⟨!ELEMENT bib (book*,article*)⟩
⟨!ELEMENT book (author+,title,publisher,comments*)⟩
⟨!ATTLIST book year CDATA⟩
⟨!ELEMENT article (author+,title,year?)⟩
⟨!ATTLIST article type CDATA⟩
⟨!ELEMENT publisher (name,address)⟩
⟨!ELEMENT author (firstname,lastname,office)⟩
```

Fig 4 An XML DTD for a computer science research bibliograph
图 4 计算机科学研究参考文献站点某一类 XML 文档的 DTD

为了便于理解将要被建立起来的查询语言,我们先把将要通过 WPDL 生成的 Web 页的内容和结构作一简单的描述.要生成的 Web 站点由 3 类 Web 页组成.首先是主页(起始页),它由计算机科学家及其所从属的研究领域(specialty)组成,其中计算机科学家某一研究领域的信息来源于基表 Scientist;其次,通过主页中的各研究领域项可以链接到该领域科学家的一个列表页(见下面 WPDL 对 SpecialtyPage()页的定义),包括科学家的姓名和他所从属的研究机构,所显示的信息来源于基表 ScientistInInstitution.在该类页面上,科学家的姓名是一个内部链接,所指向的页面(见下面 WPDL 对 IndividualPage()页的定义)包括此科学家个人和他所写的书的较详细的信息.而他所从属的机构是一个外部链接,指向该科学家所从属的机构的 Web 站点.IndividualPage()定义生成的是一类页面,这类页面的数据来源既有关系型数据库数据源,又有 XML 文档数据源.

构造主页的 WPDL 语句如下:

```
(1) CREATE PAGE HomePage()
(2) AS
(3)         Result=>{ResearchField=>{$specialties,
                                    SpecialtyPage(URL($specialties))}};
(4) USING
(5)         {($specialties) IN (SELECT DISTINCT specialty
                        FROM Scientist;)}
```

为了叙述方便,我们为每一行加了语句标号.根据前面介绍的语法,$specialties 被理解为绑定查询返回的元组集合的变量.上面的 WPDL 语句的执行结果可以表示成如下的 XML 文档.

```
⟨Result⟩
    ⟨ResearchField⟩
        ⟨specialties⟩AI⟨/specialties⟩
        ⟨SpecialtyPage⟩URL(“AI”)⟨SpecialtyPage⟩
    ⟨/ResearchField⟩
    ⟨ResearchField⟩
        ⟨specialties⟩DB⟨/specialties⟩
        ⟨SpecialtyPage⟩URL(“DB”)⟨SpecialtyPage⟩
    ⟨/ResearchField⟩
    ⟨researchField⟩
            ...
    ⟨/researchField⟩
⟨/result⟩
```

语句(4)、(5)是该 WPDL 语句的查询部分,查询的对象是关系型数据库 COMPUTIST 中的一个基表 Scientist.保留字 IN 右边的部分是一条标准的 SQL 语句,在实际当中可以写成显式调用访问 ODBC 或 JDBC 接口模块的

方式;IN 左边的部分是一个变量,用于绑定 SQL 语句返回的元组集合.例如,假设我们用 JAVA 语言来设计一个类方法,它通过 JDBC 访问 COMPUTIST 数据库,并返回一个 Enumeration 接口类的实例.则可以定义类方法如下:

Enumeration ClassName.methodName(String connectString,Array params)

以上 WPDL 语句的查询部分则可以显式地写成:

```
($specialties) IN
ClassName.methodName(password/login@machine:portnum:COMPUTIST,
                    "SELECT DISTINCT specialty FROM Scientist;")
```

接下来,我们介绍生成另两类 XML 文档的 WPDL 语句.由于前一类介绍得比较详细,我们只对与前一类有所不同的地方加以说明.

构造 SpecialtyPage()页的 WPDL 语句如下:

```
(1) CREATE PAGE SpecialtyPage(URL($specialties))
(2) AS
(3)   Result=>{Name=>
        {fname=>$firstname,lname=>$lastname,
          link=>IndividualPage(URL($firstname+$lastname)),
                    WorkingUnit=>{unit=>$institution,site=>$website}}
(4) USING {
(5)   ($firstname,$lastname,$institution,$website)
        IN  (SELECT firstname, lastname, specialty, institution, webSite
          FROM Scientist, ScientistInInstitution
          WHERE Scientist.firstname=ScientistInInstitution.firstname and
            Scientist.lastname= ScientistInInstitution.lastname
          GROUPBY specialty);}
```

语句(1)中的 URL()是一个 Skolem 函数,它根据参数不同生成唯一的 URL 地址.由此,该语句实际上定义了一类(而不是一个,因为变量$specialty 绑定到的是一个元组集合,而非单个值)XML 文档,其中的每一个实例属于一个研究领域.语句(5)是一个较复杂的 SQL 语句,但其方式与主页(HomePage())中的 WPDL 语句定义基本相同.

除了可以像上两例那样较自由地组织 XML 文档以外,也可以事先规定新站点的某类 XML 文档的 DTD,然后按照该 DTD 来书写 WPDL 构造语句.如第 3 类页面描述计算机科学家与其所写书的较为详细的信息,将其命名为 IndividualPage()(此处没有给出该类页面的 DTD).

构造 IndividualPage()的 WPDL 语句如下:

```
(1) CREATE PAGE IndividualPage(URL($firstname+$lastname))
(2) AS
(3)   $scientist=>{fname=>$firstname,lname=>$lastname,
            email=>$email,address=>$address,photo=>$photo};
(4)   $book=>{title=>$title,publisher=>$pubname,comments=>$comments};
(5)   Result=>{name=>$scientist,book=>$book};
(6) USING {
(7)   ($firstname,$lastname,$email, $address,$photo)
      IN(SELECT firstname,lastname, email, address, photo
      FROM Scientist);
(8)   ($firstname,$lastname,$title,$pubname,$comments)
      IN
```

```
WHERE book=>{author=>{firstname=>$firstname,lastname=>$lastname},
title=>$title,publisher=>$pubname,comments=>$comments} in S1);
}
```

语句(6)~(8)是该语句块的查询部分,语句(7)与上两例类似,实施对关系型数据库数据源的查询,而语句 8 则实施对所指定数据源(URL 地址为 S1)的 XML 文档进行查询.值得注意的是,语句(8)只采用了 AnXQL 查询中的 where 子句,而舍弃了 select 子句.这是因为这部分的目的只是到目标 XML 文档中匹配满足路径表达式的部分,然后把这些部分中的相关值分别绑定到变量($firstname,$lastname,$title,$pubname,$comments)上,形成一个元组集合,而不需要构造成 XML 属性表示.在构造页面之前(即在执行语句(2)~(5)之前),语句(7)和语句(8)返回的元组先进行连结运算(join).实际上,该连结运算完全是关系代数中的连接运算,较易实现.连结运算的结果形成新的元组集合,被传递给该 WPDL 的构造部分.

语句(2)~(5)是该 WPDL 语句的构造部分.为了使格式更清晰,先把科学家的个人信息与书籍信息分别构造成一棵子树,然后在语句(5)形成一棵总的 XTree.

借助于 XSLT(extensible style sheet language transformation[11])可以将由 WPDL 语句生成的 XML 文档转换成 HTML 的格式,以便于人们用浏览器进行浏览.XSL 文档中含有很多处理指令(processing instruction),如果 XML 文档的元素或属性符合某一模板指令,则该指令赋予(rendering)该元素与所包含的内容相应的 HTML 的表示形式.而这些指令的解析和实施由 XML 解析器程序和 XSL 处理器程序进行.从而基于 WPDL 语言,结合对 XML 文档和与其相对应的 XSL 文档的处理,就可以构造数据密集型的站点系统.由于篇幅所限,不再详述.

4 相关研究工作

本文的写作得益于研究界和产业界在 Web 方面对以下主题的研究:

· 对 XML 文档进行查询的查询语言的研究[9].与本研究最大的不同在于,它们不具备对异构数据源进行集成的功能.

· 对半结构化数据处理的研究,包括理论和应用方面[6,8,12].

· 数据库研究界对数据库技术应用于 Web 问题的研究[13],其中最重要的两个方面是对 Web 数据集成和 Web 站点建设和重构的研究.文献[14]主要是提出一种 Web 站点设计的方法论,设计了两种语言(Penelope 和 Telemaco)来实现 3 个建站任务的分离.然而文献[14]的研究工作只是面向关系型数据源,并且它没有形式化的数据模型支持,难以实施站点的完整性约束检验.本文的研究工作与文献[15]的出发点类似,但采用与其完全不同的技术路线.文献[15]提出的语言,查询部分和构造部分都采用特别的语法,不够直观,理解起来非常困难;而且整个站点的构成,放在一个文件或一个语句中来实现,显得缺乏灵活性.

5 结束语

本文提出了一种 Web 页面的形式化定义模型 XTree,并在此基础上设计了一种 Web 页面的定义语言 WPDL.使用 WPDL 可以将数据密集型 Web 站点建设的 3 个方面彼此分离开来,从而使站点的重构、重用和优化成为可能.

在下一步的研究工作中,我们准备进一步丰富 XTree 的语法的表达能力.WPDL 的特性也有待进一步扩展,以支持更丰富的 Web 页特性,如对具有交互能力的表单的支持等.

References:

[1] Clark, J. XSL Transformations (XSLT) Version 1.0. World Wide Web Consortium, 1999. http://www.w3.org/TR/WD-xsl.

[2] Bos, B., Lie, H.W., Lilley, C., *et al*. Cascading Style Sheets, Level 2 (CSS2) Specification. 1998. http://www.w3.org/TR/REC-css2.

[3] Fernandez, M., Florescu, D., Levy, A., *et al*. Verifying integrity constraints on Websites. In: Dean, T., ed. Proceedings of the 16th International Joint Conference on Artificial Intelligence. Stockholm: Morgan Kaufmann, 1999. 614~619.

[4] Perkowitz, M., Etzion, O. Adaptive Websites: an AI challenge. In: Dean, T., ed. Proceedings of the 15th International Joint Conference on Artificial Intelligence. Nagoya: Morgan Kaufmann, 1997. 16~23.

[5] Bray, T., Paoli, J., Sperberg-McQueen, C.M. Extensible Markup Language (XML) 1.0 Specification. 1998. http://www.w3.org/TR/REC-xml.

[6] Abiteboul, S. Querying semi-structured data. In: Afrati, F.N., Kolaitis, P., eds. Proceedings of the 6th International Conference on Database Theory. Delphi: Springer, 1997. 1~18.

[7] Buneman, P., Davidson, S., Hillebrand, G., *et al*. A query language and optimization techniques for unstructured data. In: Jagadish, H.V., Mumick, I.S., eds. Proceedings of the 1996 ACM SIGMOD International Conference on Management of Data. Montreal: ACM Press, 1996. 505~516.

[8] Buneman, P. Semistructured data . In: Proceedings of the 16th ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems. Tucson: ACM Press, 1997. 117~121.

[9] http://www.w3.org/Tands/QL/QL98.

[10] Buneman, P., Libkin, L., Suciu, D., *et al*. Comprehension syntax. SIGMOD Record, 1994,23(1):87~96.

[11] http://www.w3.org/TR/xslt.

[12] Arocena, Grustavo, Mendelzon, Alberto. WebOQL: restructuring documents, databases and Webs. In: Proceedings of the 14th International Conference on Data Engineering. Orlando: IEEE Computer Society, 1998. 24~33.

[13] Florescu, Daniela, Levy, Alon, Mendelzon, Alberto. Database techniques for the World Wide Web: a survey. ACM SIGMOD Record, 1998,27(3):59~74.

[14] Atzeni, Paolo, Mecca, Giansalvatore, Merialdo, Paolo. To weave the Web. In: Jarke, Matthias, Carey, M.J., Dittrich, K.R., *et al*., eds. Proceedings of the 23rd International Conference on Very Large Data Bases. Athens: Morgan Kaufmann, 1997. 206~215.

[15] Fernandez, M., Florescu, Daniela, Kang, Jaewoo, *et al*. Catching the boat with strudel: experiences with a web-site management system. In: Haas, L.M., Tiwary, Ashutosh, eds. Proceedings of the ACM SIGMOD International Conference on Management of Data. Seattle: ACM Press, 1998. 414~425.

An XML-Based Web Page Definition Language*

LI Xiao-dong

(*Institute of Software*, *The Chinese Academy of Sciences*, *Beijing* 100080, *China*)

E-mail: xiaodong_li@hotmail.com

http:// www.ios.ac.cn

Abstract: Data-Intensive Web sites refer to the Web sites which integrate data from multiple heterogenous data sources. Building data-intensive Web sites is a data-management problem. Generally speaking, this process consists of three main programming tasks: accessing and integrating the data available in the site, building the structure of the site, i.e., specifying the data in each page and the links between pages, and generating the HTML representation of pages. In this paper, a Web page definition language named WPDL is proposed based on XML techniques and related recommendations from W3C. A brief EBNF grammar rule of WPDL is given, and the key features of WPDL are described with a couple of examples. It is argued that the three tasks can be separated with that kind of declarative query language. Thereby it is possible to maintain a data-intensive Web site with restructuring, reusability and integrity constraint enforcement.

Key words: declarative query language; data-intensive Web-site; XTree; WPDL; XML; XSL

* Received March 9, 2000; accepted July 5, 2000