

对象模式到关系模式转换的软件框架研究

张雪东, 王淮生

(安徽财经大学 信息工程学院, 安徽 蚌埠 233041)

摘要:数据在信息系统中的使用面临着这样一个困难,即数据存储是以关系模型为基础,而软件开发以对象模型来进行,造成了软件开发中数据访问技术的不和谐。在软件实现上陷入两种模式的转换工作,破坏面向对象语言的面向对象性,造成开发效率低下,代码重用率变低。提出并建立了一个软件框架,利用它提供的处于关系数据库和客户端之间的 API 来进行基于对象的数据访问,充分发挥两种不同模式的优点,以提高软件开发的效率。

关键词:关系模型;对象模型;软件框架

中图分类号:TP311

文献标识码:A

文章编号:1673-629X(2007)11-0128-03

Research of Software Framework of Transforming from Object Model to Relational Model

ZHANG Xue-dong, WANG Huai-sheng

(School of Information Eng., Anhui University of Finance & Economy, Bengbu 233041, China)

Abstract: The using of data in information system is confronted with such a difficulty that data storage is based on relational model while software development is under object model. It makes a disharmony between software development and data access. It causes software realization change from one model to the other, destroys object-oriented character of the object-oriented language, and brings about the poor efficiency and code reusing. Here brings up a software framework, in which API between relational database and client can access relational database based on object model. The framework elaborates the merits of two models fully and improves efficiency of software development.

Key words: relational model; object model; software framework

0 引言

数据的管理大致涉及两个方面:数据存储和数据在应用系统中的使用,即数据的存取。关系型数据库由于理论完备、产品成熟、数据管理高效而在商业应用中占据着绝对的主导地位。随着软件开发技术的发展,面向对象的软件开发方法广泛应用,面向对象的软件建模已经成为构架软件最流行和快捷的方法。这种状况导致软件开发中数据访问技术的不和谐:既要适应分析和设计阶段的面向对象性,又得在关系模型下进行数据管理,在编码上陷入两种模式的转换工作,破坏面向对象语言的面向对象性,造成开发效率低下。目前,面向对象数据库(OODB)作为一种新型数据库技术并没有取得商业上的成功,因为它要求已有数据转换为 OODB 格式,转换成本十分昂贵^[1]。文中提出

并建立了一个软件框架,它可以较好地解决上述问题。

1 软件框架的提出

软件框架结构如图 1 所示,通过提供一个实现在数据库访问部分的 API 库,来辅助和简化由 OOA 和 OOD 构造起来的系统在数据库访问上的编程,使得新系统和原有系统可以很好地共存,数据无需迁移,可以大大节省企业的再次投资。在图中,虚线上方是关系型数据管理服务及已有的数据访问驱动程序,它提供的是关系模式下的数据管理。虚线下方就是软件框架所提供的功能模块,它总的功能就是在现有关系型数据服务的基础上给开发者一个面向对象的数据管理视图^[2]。

在软件框架中,数据库访问层接受模式映射层生成的数据库访问语句,通过现有数据访问驱动访问数据库,同时接受数据库返回的结果给模式映射层,完成数据传送任务;模式映射层的主要任务是模式转换^[3],即将辅助编码层传递来的对象转换为关系型数据库所

收稿日期:2007-02-10

基金项目:安徽省教育厅自然科学研究项目(2005kj052)

作者简介:张雪东(1980-),男,安徽蚌埠人,硕士,讲师,研究方向为数据库技术、算法研究。

能存储的数据结构,同时要为辅助编码层提供对象组装的有效信息;辅助编码层提供面向对象的数据访问 API,数据的增加、删除和修改在编程者看来完全是应用层数据型对象的行为的调用,具体的数据增加、删除和修改功能由软件框架提供^[4]。

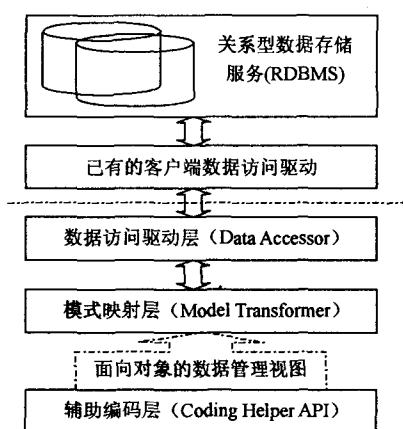


图 1 软件框架模型

2 软件框架的设计与实现

2.1 辅助编码层

辅助编码旨在为应用层编程提供一个面向对象的数据访问视图,它包含一个数据型对象基类,所有应用层对象的类都继承于它,基类提供从数据库初始化、在数据库中删除指定对象、插入新对象到数据库和在数据库中更新对象等操作。

数据型对象基类把要转化的对象的类结构信息(包括关联信息、聚集信息)、类关系信息以及继承关系所形成的类层次信息注册到映射信息类,同时给出数据访问的接口,用于辅助编码,使应用系统编码人员以对象模型的视图管理数据。数据型对象基类结构如下:

```
class DataObj{
    Classname; //类名
    pTransInfo; //指向映射信息的指针
    .....
    InsertNew(); //新对象保存
    DeleteSelf(); //对象删除
    UpdataSelf(); //对象更新
    GetSelf(键值); //初始化对象
    DoReg(TransInfo * p_info); //信息注册函数
    .....}
```

2.2 模式映射层

模式映射层接收应用层提供的信息,包括类结构信息、类和类之间的关系信息、对象之间关系信息、对象的操作动作序列信息,这些信息按照一定的规则定义,模式映射层分析解读信息,生成数据库访问语

句^[5]。模式映射层涉及到的数据结构有映射信息类、访问参考信息、访问请求队列。模式映射层结构图如图 2 所示。

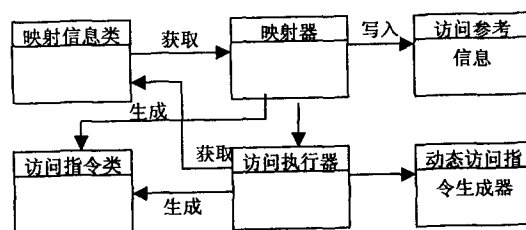


图 2 模式转换层结构图

(1) 映射信息类(TransInfo)。

应用层数据型对象的静态信息都由映射信息类来表示,在映射信息类里面要完整描述待存取的应用层类层次信息,涉及类结构、类继承、静态关联和组合等信息,如图 3 所示。

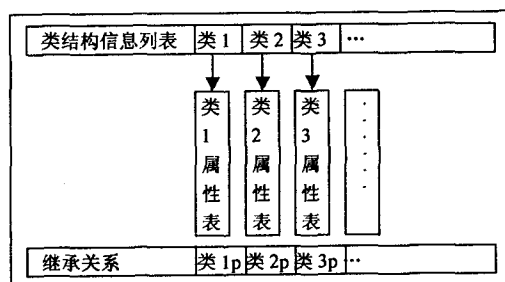


图 3 映射信息类示意图

① 类结构信息列表:保存各个类的名称及指向该类属性表的指针。

② 继承关系:假设继承关系为单重继承(多重继承可以转换为单重继承),在继承关系示意图相应位置只要填入类结构信息列表在该位置的类的父类的标识即可。

③ 类属性表:记录类的每个属性的类型、名称、与该类实例的关系(表述关联关系的类型)以及实现方式(指对象之间的关联关系的引用方式,可以用指针指向,也可以是整个对象的拷贝)。

映射信息在客户端保存,系统运行时需要初始化好映射信息,以便在系统运行时参照,具体有映射器映射函数在模式映射时和访问执行器在对象组装时参考。

(2) 映射器(Transformer)。

映射器获取要翻译的类映射信息(源数据),把映射信息转换为目标指令和访问规则(目标数据),把访问参考信息写入规则表,调用访问执行器执行数据访问指令。映射器类的设计如下:

```
class Transformer{
    属性:
    pTransInfo; //指向要翻译的映射信息的指针
```

pRef; //指向访问参考信息的指针

pInstructionList; //指向生成的目标指令列表的指针

方法:

Trans(); //模式映射函数

GenInstList(); //生成指令列表

CanBeAppend(); //判断是否可以 Append 方式修改数据库

.....|

(3)访问执行器(AccessExecutor)。

访问执行器传送访问指令给访问层,以及存取访问层的结果缓冲,并且执行对象的组装,返回结果给应用层。AccessExecutor 可以由辅助编码层中数据型对象类的实例调用,这种方式多是应用层的动态请求,需要调用动态访问指令生成器来把请求转换为访问指令。第二种调用是由映射器调用,这种方式是应用层数据型对象静态信息的访问请求,指令已由映射器生成好。访问执行器类设计如下:

```
class AccessExecutor{
```

属性:

CurrentMode; //当前访问模式,分为方式一,方式二

pInstructionList; //访问指令表指针

pRef; //指向访问参考信息

pTransInfo; //映射信息

方法:

ReqToInstruction(); //访问请求转换为指令的方法,调用动态指令生成器

Validate(); //访问验证

AccessExe(); //执行访问

|

(4)访问参考信息。

由模式映射器根据映射信息生成,作为访问执行器的访问参考,其具体的生成细节完全由映射理论中访问约束而定^[3]。

(5)动态访问指令生成器类。

根据对象发出的数据库访问请求生成访问数据库的指令。

2.3 数据访问层

数据访问层的任务就是接收访问指令,访问数据库,缓冲访问结果,传递结果信息。数据访问层接受模式映射层生成的数据库访问语句,在现有数据访问驱动(如:ODBC,JDBC,ADO 等)之上完成数据访问任务。构建新的数据访问驱动的目的是为了设计的可扩展性与结构的一致性,同时给模式映射层一个友好的访问方式。数据访问层结构如图 4 所示。

(1)访问管理类。

包含数据访问驱动层的主线程,负责这一层的对象之间的调度和与上下层的交互,其算法流程如图 5

所示。

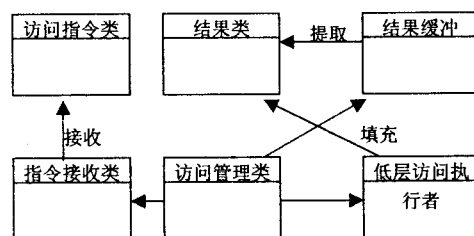


图 4 数据访问层结构图

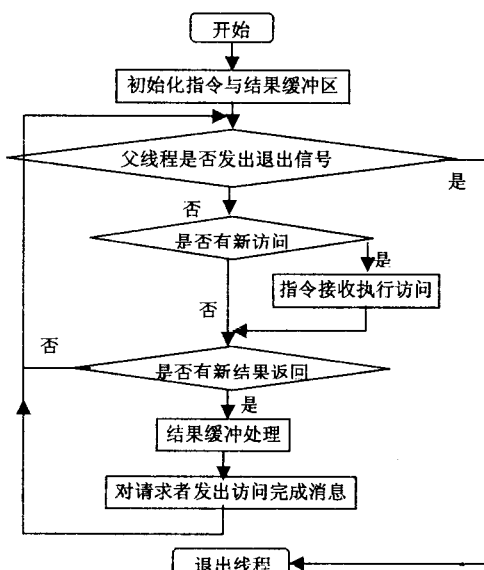


图 5 访问管理者算法流程图

(2)访问指令类(Instruction)。

数据访问层与模式映射层相互配合具体体现在接受指令的格式和返回结果的表示上,指令类设计为:

```
class Instruction{
```

属性:

Statement; //指令内容

Sender; //指令发出者

Type; //指令类型

Id; //指令编号

ParentId; //父指令编号,访问分解时使用

方法:

GetStatement(); //获得指令内容

SetStatement(); //设置指令内容

GetParentId(); //获得父指令编号

GetId(); //获得指令编号

.....|

(3)结果类。

返回结果在数据访问层采取半格式化的方式,对于只要获取操作成功与否的返回结果,用一个类来表示,该类包含了操作的发起者、操作结果、操作内容等信息。对于要获取一个对象数据的操作,由于低层驱

(下转第 134 页)

Transforms); 标明此元素应交给哪个业务方处理, 用业务方的代号来表示 To 属性的值。设计 DigestValue 元素。

(2) 将各个 SubDigest 元素放在一起, 生成 SubDigests 元素。

(3) 设计 ds:Signature 元素(其中, 用 ds:Reference 元素来表示需要连接的所有的子摘要和联接摘要值; 用 ds:SignatureValue 元素表示整体签名值)。

(4) 设计 UnitedSignatureObject 元素。用来放置其他信息, 包括数据对象、时间戳等。

(5) 设计 UnitedSignature 元素。

3.3 XML 分别验证实现过程

各消息接收方收到 XML 整体签名文档以及自己感兴趣的信息, 根据 XML Schema 分别验证自己接收数据的完整性和发送者身份认证。具体实现过程:

(1) 从 SubDigests 元素中通过 SubDigest 元素的 To 属性值找到本业务方应处理的那些 SubDigest 元素。

(2) 通过 SubDigest 元素的 DigestInfo 子元素的 DigestReference 元素得到被计算摘要的数据对象。其间, 要通过 DigestReference 的 URI 属性表示的数据对象引用和子元素 ds:Transforms 元素表示的转换列表得到目标数据对象。

(3) 根据 Digest 元素所给出的规范化算法规范化上面所得的数据对象, 并根据 Digest 元素给出的摘要算法计算规范化后的数据对象的摘要值。

(4) 用此摘要值替换 DigestValue 元素的内容。

(5) 验证 ds:Signature 元素。

(上接第 130 页)

动程序返回的只是记录, 需要对它们进行再处理, 因此, 还需要缓冲发出数据访问的对象的信息, 以及其他一些额外的信息, 为应用层数据型对象在模式映射层的组装预备了必要的信息, 这个过程称为数据访问结果的半格式化。

(4) 低层访问执行者。

低层访问执行者接收 SQL 语句^[6], 访问数据库, 并把结果半格式化到结果缓冲。

3 结束语

对象模式到关系模式转换的软件框架给开发者一个面向对象的编程视图, 利用软件框架提供的 API 可以对关系数据库进行应用层对象的直接存取, 大大提高了软件开发效率, 并可以在对数据不迁移的情况下

4 结束语

设计实现的一种改进的 XML 签名技术具有整体签名、分别验证的功能, 解决了基于 XML 通信技术的业务链中多方通信时的身份认证和 XML 数据完整性、交易防抵赖的保护。它所具有的技术优势使其在以电子商务为代表的网络应用中有很好的前景。

参考文献:

- [1] Harold E R. XML 宝典[M]. 马云, 钟萍等译. 北京: 电子工业出版社, 2002.
- [2] W3C. XML - Signature Syntax and Processing[EB/OL]. 2002-02-12. <http://www.w3.org/TR/xmldsig-core/>.
- [3] Mertz D. 密码学简介: 第一部分[EB/OL]. 2002-04-26. <http://www-900.cn.ibm.com/developerWorks/cn/cnedu.nsf/security-onlinecourse-bytitle/>.
- [4] Mertz D. 密码学简介: 第二部分[EB/OL]. 2002-05-17. <http://www-900.cn.ibm.com/developerWorks/cn/cnedu.nsf/security-onlinecourse-bytitle/>.
- [5] Mertz D. 密码学简介: 第三部分[EB/OL]. 2002-05-24. <http://www-900.cn.ibm.com/developerWorks/cn/cnedu.nsf/security-onlinecourse-bytitle/>.
- [6] XML Schema Part 0: Primer. W3C Recommendation[EB/OL]. 2001-05-02. <http://www.w3.org/TR/xmlschema-0/>.
- [7] XML Schema Part 1: Structures. W3C Recommendation[EB/OL]. 2001-05-02. <http://www.w3.org/TR/xmlschema-1/>.
- [8] XML Schema Part 2: Datatypes. W3C Recommendation[EB/OL]. 2001-05-02. <http://www.w3.org/TR/xmlschema-2/>.

对系统进行升级。

参考文献:

- [1] Johnson J L. Database: Models, Languages, Design[M]. 北京: 电子工业出版社, 2004, 5.
- [2] 赵致格, 王枯民, 雍俊海. 面向对象数据库中对象的存储和操作算法[J]. 计算机辅助设计与图形学学报, 1998, 10(1): 15-21.
- [3] 王学荣, 曾晓勤. 从面向对象数据库模式到关系数据库模式的转换[J]. 计算机工程与科学, 2003, 25(5): 100-107.
- [4] 毕南楠, 方昌始. 一种新的面向对象数据库建模框架[J]. 计算机与数字工程, 2006, 34(7): 67-70.
- [5] Blaha M, Premerlani W, Shen H. Converting OO Models into RDBMS Schema[J]. IEEE Software, 1994(1): 28-39.
- [6] 王学荣, 曾晓勤. 面向对象数据库的查询转换成关系数据库的查询[J]. 计算机工程与应用, 2002(20): 176-182.