用 Java 来设计组件重用的 Query 方法

葛瀛龙　徐 翀　郑　宁　胡昔祥
(杭州电子工业学院,杭州 310037)
E-mail:geyl@fulltop.com

摘　要　文章所探讨的组件重用的 Query 方法是利用 Java 反射技术和 Java 数据库连接技术来完成对数据库的查询操作。它将查询数据库的公共操作封装于一个组件中,使编程工作者在具体编程工作中能方便地重复使用它们,以求简化编程工作。

关键词　EJB　Java 反射技术　Java 数据库连接

文章编号 1002-8331-(2002)20-0103-04　**文献标识码** A　**中图分类号** TP311

Design a Component Reused Query Method by Java

Ge Yinglong　Xu Chong　Zheng Ning　Hu Xixiang
(Hangzhou Institute of Electronics Engineering, Hangzhou 310037)

Abstract: For simplifying the programming work, this paper discusses a component reused query method which is designed by Java reflection technique and Java database connection technique. The method encloses the common operations that query the database into a component, so that the programmer can reuse them conveniently in the programming.

Keywords: EJB, Java reflection technique, Java database connection technique

随着 Internet/Intranet 的进一步普及,网络应用的增加,编程业务量越来越大。基于 WEB 技术构建的三层或多层模型体系结构较好地解决了网络编程中的许多问题。然而这种多层模型体系的应用程序通常较难编写,因为它们要调用许多复杂的代码来完成事务处理、状态管理、多线程、资源池和一些底层工作。除此之外,每编写一个新应用程序时,程序员还必须重写这些复杂的代码。若程序员能够使用预先编译并且测试好的事务管理代码,或是某些自己已写好的代码进行重用,那么它可以节省大量的时间和精力。该文采用了类似于 EJB 中的 Finder 方法的功能,较好地实现重用代码的目的。

1　EJB 简介

1.1　EJB 框架概述

Sun 公司的 Enterprise JavaBeans(EJB)技术向程序员提供了一种能极大提高代码复用的机制。EJB 是一个开发和部署分布式服务器端的、带事务处理的、安全的商业组件的规范和结构。图 1 描述的是 EJB 组件的多层体系结构模型图。

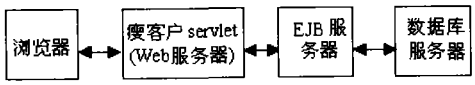

图 1　Enterprise JavaBeans 组件多层体系结构模型图

当今较为流行的三层或多层体系结构是对标准两层客户/服务器模型的扩展,它通过在客户端和数据库之间增加一个多线程应用服务器而实现。EJB 就在这个服务器上执行。

1.2　EJB 中的 Finder 方法

EJB 组件中最基本的两类组件为:Session bean (会话组件)和 Entity bean(实体组件)。Session bean 用于处理业务逻辑。Entity bean 代表业务处理对象,一个简单的实体 bean 可以定义成代表数据库表的一个记录,也就是每一个实例代表一个特殊的记录。更复杂的实体 bean 可以代表数据库表间关联视图。每一个实体 bean 都必须有一个主键值来唯一标识它。每个实体 bean 的接口提供了 Finder 方法。它相当于数据库中的 select 功能(查询与检索)。Finder 方法允许客户端查找 Entity bean,它的最基本的两个查找方法是 findByPrimaryKey(靠关键字来查找记录)和 findAll(查找数据库表中所有记录)。一个实体 bean 的缺省的 find 方法是 findByPrimaryKey 方法。它通过主键来定位实体 bean。每一个实体 bean 必须实现 findByPrimaryKey 方法。其语法如下:

<remote interface> findByPrimaryKey(<key type> key)

有可能一个 Finder 方法得到满足 SELECT 语句条件的多个记录,此时它返回一个包含多个实例的对象集。

1.3　中小型系统中 EJB 开发的局限性

EJB 体系结构对于应用开发者提供了下列好处:应用移植性、组件重用性、构件复杂应用的能力、业务逻辑与呈现逻辑的

基金项目:国家教育部优秀青年教师基金资助项目

作者简介:葛瀛龙,男,研究生,主要研究方向为电子商务、电信网管。徐翀,女,研究生,主要研究方向为电信网管、电子商务。郑宁,男,杭州电子工业学院 CAD 所研究员、教授、硕士研究生导师,主要研究方向为 ICCAD、电信网管、电子商务。

分离、可部署在多种操作系统环境中、分布式部署、应用互操作性、与非 Java 系统的集成。

虽然 EJB 有以上的多方面好处，但不可避免地也有许多局限性：

(1)对一般的 Java 程序员而言，要理解 EJB 的体系结构，尚需更多的时间。在理解它的体系结构后，还需要对分布式计算及 CORBA 有一定的了解，这样就延长了开发期。

(2)要开发稳定的和可移植的 EJB 组件，就必须遵守 EJB 的多个限制条件，这样便束缚了程序员的手脚(比如进行较为复杂的数据库查询操作，及其它的备份还原操作时，仅有的 EJB 方面的组件及 API 就显得力不从心了)。

(3)在开发一些简单的中小型项目、一些涉及分布式计算较少的系统时，EJB 的优点就无法体现出来。相反，它会增加系统的开销，降低系统运行的效率。

2 基本设计思想

在 EJB 技术中最值得称道的就是其中之一的组件重用的 Finder 方法，它屏蔽了底层数据库连接的一些细节，程序员不需要知道太多的 sql 语句就可进行数据库检索的操作；它还封装许多数据库的底层操作(如事务处理，数据库打开关闭操作等)。如果程序员在设计中小型的系统时，想要使用 EJB 中的 Finder 方法来省去许多重复的操作，而又不涉及到复杂的 EJB 技术，那么能否设计一种脱离 EJB 体系结构的新 Finder 方法呢？该文提供了设计这种基于服务器的组件重用的 Query 方法的思路。如果不采用组件重用的 Query 方法来编写查询操作，那么一般需要经过以下五个步骤才能完成数据库的查询操作：

(1)初始化，设定各种变量初始值；

(2)生成 sql 语句；

(3)打开数据库连接并发送 sql 语句；

(4)处理返回的结果集；

(5)关闭数据库连接，返回最终的结果。

然而在以上的五个步骤中，第(3)、(4)、(5)步在不同的查询方法中都是重复出现的，它们可以被写在一个公共的方法中，第(2)步可以通过 Java 的反射技术构造出一个公共的模型来适应不同的查询方法的需要。该文提出的这种基于服务器的组件重用的 Query 方法就是遵循这种思路来实施的。Query 可以包括 queryall 方法、queryByPrimaryKey 方法 (根据关键字查询)、queryByOthers 方法(根据组合项查询)等。各种 Query 方法的基本思路是一致的。

3 Query 方法及实现

3.1 创建前提条件

在实现 Query 方法之前，必须进行以下步骤来获得 Query 方法所需要的各种参数：

(1)底层模型 Bean 的结构。

该文所论述的模型 Bean(Model Bean)，类似于 EJB 中的实体 Bean。一般来说，一个模型 Bean 关联于数据库中的一张表。它的结构如图 2(以管理员类的模型 Bean 为例)。

模型 Bean 中包含了各种构造及描述该 Bean 的方法：构造方法、读写属性的方法(getter and setter)、映射表方法、创建数据库实例的方法(creatNewModel)等。

(2)建一张映射表。

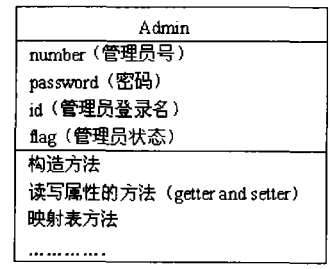

图 2 模型 Bean 类图

每个实体 Bean 都有一张相应的表，该表用来存放数据库字段到实体 Bean 属性的映射，一般使用哈希表的技术来创建相关映射。例如：(用管理员表作为例子)

sql 语句创建表 create table Admin (NUM_ID int(4)NOT NULL UNIQUE PRIMARY KEY,

NAME_TX char (20)NOT NULL,PWD_TX char(20)NOT NULL,

FLAG_NR int(4)NOT NULL);

相对应的模型 Bean 的属性名为(Number,Id,Password,Flag)。

在它们之间用哈希表建立对应关系。当要通过数据库字段来获得模型 Bean 的属性名时，可以使用方法：

属性名 =(String)映射表.get(数据库字段)；

(3)对模型 Bean 中的各种方法分类。

通过对命名的约定(如 setter 方法的方法名的前三个字母是“set”)，从模型 Bean 的接口中获得该 Bean 的各种方法名，包括读属性、写属性、创建一个模型 Bean 和操作映射表的方法名。然后利用 Java 的反射技术来获得模型 Bean 中的相关方法。Query 方法会使用其中一些方法来读取属性和修改属性，可以参见图 3 的过程描述：

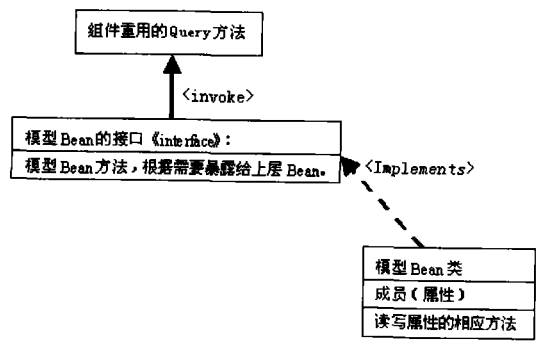

图 3 Query 方法提取模型 Bean 的方法

(4)处理结果集的构造方法。

在模型 Bean 相应的接口中有一个处理数据库结果集的方法，该方法将得到的结果集中的对象封装到一个新的模型 Bean 中。下面将以管理员的模型 Bean 作为例子：

……

管理员号 = new Integer(rs.getInt("NUM_ID"));//rs 为数据库返回的结果集

管理员登录名 = rs.getString("NAME_TX");

管理员密码 = rs.getString("PWD_TX");

管理员状态标记 = new Integer(rs.getInt("FLAG_NR"));

……

return new 管理员模型 Bean 的构造函数(管理员号、管理员登录名、管理员密码、管理员状态标记);

(5)封装必要的数据库操作方法。

将必要的数据库操作方法封装在一个类中以方便重用,其中包括打开关闭数据库、执行查询、执行更新、批操作和设置自动提交等。一些数据库的驱动程序及 URL 写入一个配置文件中,不但可以提高重用性,而且更改驱动程序及 URL 时,不必重新编译程序。

3.2 具体实现

首先,需要对相关的变量进行初始化,建立接口单元,一些存储单位存放接口中读方法集(getter)、写方法集(setter),创建映射表的方法单元等。

进行初始化完毕后,利用反射获得接口中的所有方法,存入接口方法集中。

```
……
//ifmethods 中包含模型 Bean 提供给上层的接口中的方法
Method[]ifmethods=模型 Bean 所对应的接口.getMethods();
……
```

从这些获得的方法中,根据命名的约定(前提条件 3)分离出读属性方法(getter)、写属性方法(setter)和操作映射表的方法。

其次,获得了所需要的方法后,就可以生成相应的 sql 语句,这是最关键的一步。但是对于不同的 Query 方法,生成的 sql 语句都不同。比如较为简单的 queryAll 方法所生成的 sql 语句为:

```
sql="select*from"+数据库表名;
```

而其它的 Query 方法相对就要复杂些。其中最复杂的 Query 方法是组合查询 queryByOthers(依据数据库中的任意几项进行查询),这就需要利用前提条件 2。映射表的映射方法需要用户提供依据项(依据的属性)的数据库字段集。通过该字段集,在映射表中找到对应的属性名,然后获得它们的读写方法名。利用反射技术调用对应的读方法获得属性值,最后构造 sql 语句。例如,需要构造以下一种 sql 语句:

```
sql="select * from"+数据库表名+"where"+数据库字段名(依据项)1+"="+模型 Bean 中对应的读方法(getter)获得的属性值 1+"and"+数据库字段名 (依据项)2+"="+模型 Bean 中对应的读方法(getter)获得的属性值 2+"and"+……;
```

可以从以上的 sql 语句抽取它们的共性来获得以下的通用过程:

```
……
for(int i = 0;i <对应的依据项的读方法个数;i++){
……
```

在模型 Bean 所对应的接口中,提取对应依据项的读方法(setter 方法);

```
属性值 = 依据项的读方法.invoke (模型 Bean 的实例, null).toString();
}
```

然后利用以下方式构造 sql 语句:

```
String result=依据项的数据库字段名+"="+得到的依据项的属性值+"";
for(int j = 1;j < 依据项的个数-1;j++){
tempstr ="and"+依据项的数据库字段名+"="+用读方法(getter)获得依据项的属性值+"";
result = result + tempstr;
}
sql="select*from"+数据库表+"where"+result;
……
```

然后,打开数据库连接,发送 sql 语句,该过程需要使用先决条件 5 中提到的数据库操作:打开数据库连接 ----> 创建陈述集(Statement)----->发送 sql 语句。数据库接受 sql 查询语句,并执行 sql 查询语句,返回 sql 语句执行后的结果集。而处理结果集的过程是利用 Java 反射技术调用对应模型 Bean 的结果集构造方法(前提条件 4),将返回的结果集中的值封装成一个对应的模型 Bean 的新实例,最后返回这个新实例或由多个新实例组成的 Collection 集。

最后要完成的步骤是关闭数据库连接,返回结果。上层 Bean(数据库层)调用查询方法时,只需写 3~5 行代码就可以完成以前需要写 40~50 行代码的工作。用 Query 方法获得查询的结果,对需要的结果进行恰当的造型。然后就可以对造型后的结果进行操作,调用其读属性的方法,获得属性值提供给客户端使用。以下以处理组合查询 queryByOthers 的返回结果为例:

```
ArrayList al = new ArrayList();
//设定依据项在数据库中的字段名(该例是根据名字和密码查找)
String[]items={DISK_ADMIN_NAME,DISK_ADMIN_PWD};
al=(ArrayList)DBCommon.queryByOthers (数据库表名, items,传入的实体 Bean 的接口);
//对获得的结果恰当的造型
return(传入对象的类)al.get(0);//返回结果集中第一条纪录
}
```

4 Query 方法所使用的关键技术

文中的 Query 方法是使用网络编程语言 Java 完成的,它主要涉及到了基于 Java 的三个重要技术:Java 的反射技术实现对对象原类及原接口的访问;JDBC 实现对数据库的读写操作;Java 的接口技术。其中,反射技术是该方法的核心技术。

4.1 Java 的反射技术实现对对象原类及原接口的访问

Java(TM)的反射技术可使 Java 代码发现有关已加载类的域、方法和构造函数的信息,还能用被反射的域、方法和构造函数在安全限制内对其基本对应的对象进行操作。该次设计中自始至终都用到了 Java(TM)的反射技术。Java(TM)的反射技术核心 API 定义了三个主要的类 (Field、Method 和 Constructor),它们分别代表了反射的类中的域、方法和构造函数。其中 Method 类中的 invoke 方法会调用反射类所指定的方法。运用这种反射方法的显而易见的好处就是:当后期向数据库增加内容时,只需要程序员修改相应的模型 Bean 或代价较小的业务逻辑就可以完成修改工作。因为反射技术会自己查找各种方法,Query 方法会自动完成对数据库方面的操作。

4.2 用 JDBC 实现对数据库的读写操作

Java 语言用 JDBC 驱动程序来访问数据库,它定义了一共 4 种类型的协议标准,各数据库厂商均有其各自的实现办法。该次设计中使用了 JDBC-ODBC Bridge 的驱动程序。采用 JDBC 定义的 API 对象和方法,可以与底层数据库交互。通过 URL 识别数据库,建立与数据库的连接,生成 sql 语句对象,传

表 1 实验数据比较表

查询方法 / 比较项目	EJB 中的 Finder 方法（一般由 EJB 容器管理）	该文所论述的 Query 方法	一般的数据库查询方法
数据库层代码量	只定义了接口，由 EJB 容器实现，有较多的限制条件	3~5 行	40~50 行
数据库对象层代码量（以一个查询方法为例）		80 行左右	70 行左右
易用易学性	较难理解，不易使用	最容易学习和使用	较容易学习和使用
数据库表结构变动时，代码修改情况	需修改实体 Bean(接口都要更改)	只需少量的修改数据库对象层	数据库对象层和数据库层均需要修改
运行效率	由容器决定	较快	快

送 sql 语句到底层的数据库，执行 sql 语句，即可访问数据库。该文所采用的设计思想是覆盖所需的数据库操作方法，抽出它们的共性并形成一个类。如 DatabaseConfig 用来读写数据库配置文件，包括驱动程序，URL 等；而 DBHandler 用来管理数据库的打开、关闭、创建 Statement 等。

4.3 使用 Java 的接口技术向上层提供相应的读写方法

接口技术也是该次设计中的关键技术之一，它符合 MVC (Model-View-Control，模型-视图-控制)的设计模式。Model 就代表文中所提出的模型 Bean，它可以有多个接口(相当于视图 View)，不同的视图可以提供给不同的业务逻辑。文中 Query 方法需要一些 Model 中的方法，那它就可以形成它的 View，而这些 View 仅仅包含了 Query 方法所需要的函数。这种 MVC 的结构可以见图 4。

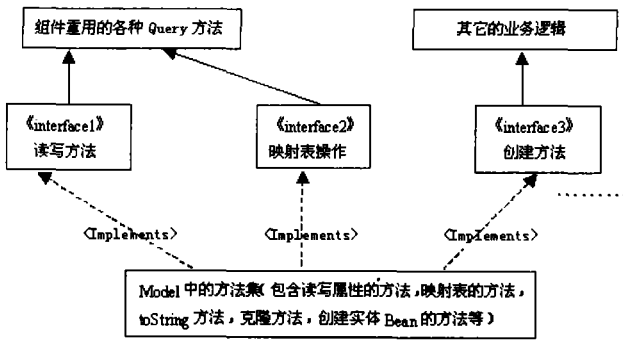

图 4 符合 MVC 设计模式的 Query 方法

5 实验数据比较

表 1 是基于管理员数据库为例进行比较。硬件环境是：Intel Pentium Ⅲ 466，256M 内存，8GB 硬盘；软件环境是：Windows 2000 Advanced Server，Sun J2EE Server，JBuilder 5 开发工具，MS SQL Server2000 数据库管理系统。表 1 将三种查询方法进行了比较。

6 结论

文章的 Query 方法采用了 Java 反射技术，接口技术，数据库连接技术等。适用于中小型的项目。采用该方法编程，程序员的代码量会显著的减少，工作效率会提高。在表 1 中，数据库对象层代码量相对其它层较为稳定，很少修改，数据库中一张表对应于一个对象。在数据库层中，程序员会为一个数据库对象写许多查询方法，如果使用 Query 方法，那么程序中每个查询方法的代码量仅为不使用 Query 方法的代码量的十分之一。当数据库表结构变动时，使用 Query 方法编写的程序较容易维护，只需要少量地修改数据库对象层代码。而不使用 Query 方法编写的程序就需要对数据库对象层和数据库层代码进行大量修改并测试，这样就增加了维护的难度。比较三种方式，EJB 最难学习且不易理解，要掌握它的 Finder 方法也较难，学用 Query 方法编写程序最容易。系统运行效率上，用 Query 方法设计的程序的运行速度几乎和不用 Query 方法设计的程序的运行速度一样快。对于用类的组合方法设计的程序，Query 方法也是适用的，只需在每个实体 Bean 中加上相应的处理方法即可。

上述的设计过程还适用于更为广泛的组件重用技术，比如数据库查询方法外的其它数据库操作方法——插入(insert)、更新(update)、删除(delete)、备份还原(backup and restore)等。它们的基本思想方法是一致的。这样的设计不但为以后的编码减轻了负担，还为后期的维护工作带来了方便。使用 Query 方法时，必须要有一套统一的命名规范，才能保证设计的正确性。

(收稿日期：2002 年 3 月)

参考文献

1.Sun Microsystems Inc.JDK1.3.1 documentation.http://java.sun.com/j2se/1.3/docs/index.html,1999-2000

2.Calvin Austin,Monica Pawlan.Advanced Programming for the Java2 Platform[M].机械工业出版社,2001

3.Vlada Matena,Beth Stearns.Applying Enterprise JavaBeans Component-Based Development for the J2EE Platform[M].机械工业出版社,2001

4.黄理，李积善等.用 JSP 轻松开发 Web 网站.北京希望电脑公司，2001

(上接 49 页)

1.Preparata F P,J S Hong.Convex Hulls of Finite Sets of Points in Two and Three Dimension[J].Communications of the ACM,1977;20(2):87~93

2.Akl G S,Toussaint T G.A Fast Convex Hull Algorithm[J].Inf Proc Lett,1978;7(5):219~222

3.Andrew M A.Another Efficient Algorithm for Convex Hulls in Two Dimension[J].Inf Proc Lett,1979;9(5):216~219

4.Preparata F P,Michael Lan Shamos.Computational Geometry:an Introduction[M].New York:Springer-Verlag,1985

5.金文华，何涛，刘晓平等.基于有序简单多边形的平面点集凸包快速求取算法[J].计算机学报，1998;21(6):533~539