J2EE 中对象关系映射的研究与实现

余俊新,孙 涌
(苏州大学 计算机科学与技术学院,江苏 苏州 215006)

摘 要:在软件开发中,对象关系映射主要用来解决对象模型到关系模型的映射问题。目前对于这个问题的解决大多是一种纯对象关系映射的实现,文中分析了这种纯对象关系映射机制所存在的不足,并针对这些不足,给出了一个半自动对象关系映射的实现。通过使用该系统,应用程序开发人员能够以一种面向对象的方式来进行对象在关系数据库中的存取,从而极大地提高了开发效率和应用程序的可维护和可扩展性。

关键词:对象关系映射;持久化;对象模型;关系模型
中图分类号:TP311 **文献标识码**:A **文章编号**:1673-629X(2007)03-0088-03

Research and Implementation of Object Relation Mapping in J2EE

YU Jun-xin, SUN Yong
(School of Computer Science & Technology, Soochow University, Suzhou 215006, China)

Abstract: Object relation mapping solves the problem of mapping between object model and relation model. Currently, most of the solution is pure object relation mapping. Analyzes the deficiency of the mechanism of pure object relatioin mapping in order to overcome the efficiency. Provides a semiautomatic implementation of object relation mapping. The developer can take an object-oriented method reading object from RDBMS and writing object to RDBMS via this system, and enhances the development efficiency and robustness of application program greatly.

Key words: object relation mapping; persistence; object model; relation model

0 引 言

在使用面向对象程序设计方法开发企业应用程序时,通常需要将一些对象保存到持久存储介质以便以后来提取或者修改,这种对象被称为持久对象[1],而相应的存储介质则被称为持久化机制,它可以是普通的文件、关系数据库或面向对象数据库。

由于面向对象模型与关系模型之间存在的区别,在将对象持久化到关系数据库的时候,会遇到两者之间的阻抗不匹配,这种不匹配使得开发人员在开发过程中,不得不付出相当一部分精力来解决非业务领域相关的问题。为了解决这个问题,通常通过一定的映射方法将持久对象存储到关系数据库中,这种映射方法就是对象关系映射[2](Object Relation Mapping),简称 ORM。

目前对象关系映射的实现大多数都是一种纯 ORM 实现,它们被称为纯对象关系映射,或者全自动对象关系映射[3],在这类实现中,它对对象到关系的映射机制进行了完整而又封闭的封装。这种全面的封装给开发者带来了很多好处,但是,在许多情况下,这种完整的封装也表现出了它的不适应的一面。基于此,文中研究了一种对对象关系映射相对来说较开放的封装,即半自动对象关系映射[3]。

1 对象关系映射概述

对象关系映射系统实际上是位于用户的商业逻辑类和关系数据库之间的一个持久层,持久层是负责商业逻辑类与关系数据库之间进行通信的一组类,它解决了对象模型和关系模型的不匹配,使得商业逻辑类与关系数据库具有无关性。

如图 1 所示,在这个持久层中有一个映射规则库,其中存放了对象到关系数据库中的表的映射信息:当商业逻辑类需要持久化时,它发出一个消息给持久层,持久层从数据库中读出该类对应的表,执行相应的 SQL 代码,从而实现对象的存取。商业逻辑类与持久层之间传递对象,持久层给关系数据库传递 SQL 语句,关系数据库向持久层传递记录。这样商业逻辑类在存取对象时,就象在对象数据库中存取一样。

收稿日期:2006-05-27
作者简介:余俊新(1981-),男,湖北荆州人,硕士研究生,研究方向为分布式计算;孙 涌,副教授,研究方向为软件工程的研究。

对象关系映射机制通常从如下三方面入手[4]:

1)类属性映射到数据库表列。

2)类映射到数据库表。

3)类间关系映射成键值。

关于对象关系映射机制的更详细介绍可以参考文献[5]。

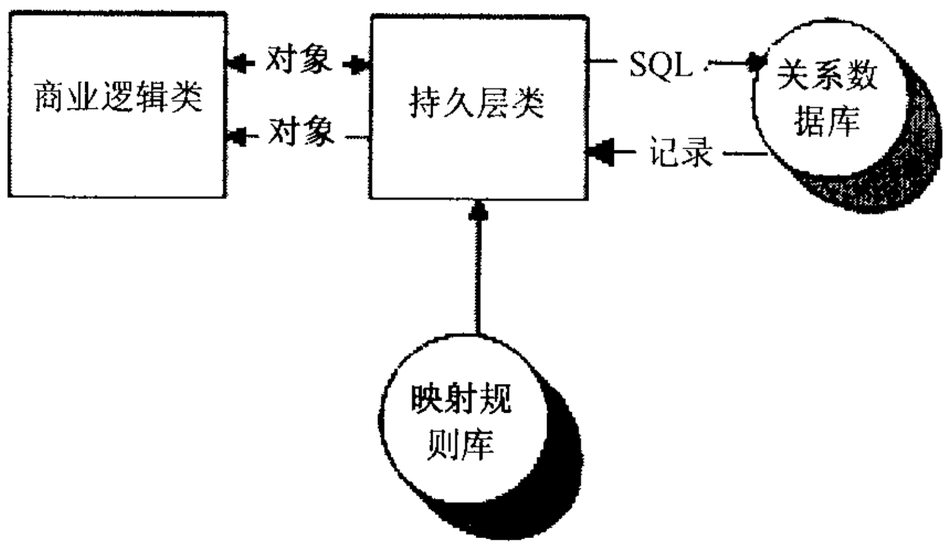

图 1　持久层

2　纯 ORM 与半自动 ORM 的对比

2.1　纯对象关系映射的特点

全自动对象关系映射的实现如 Hibernate,Apache OJB 等,它们的一个共同特点就是对数据库结构提供了较为完整的封装,提供了从对象到数据库表的全套映射机制。开发人员只需定义好从对象到数据库表的映射关系,即可通过 Hibernate 或 OJB 提供的方法来完成持久层操作。在运行时刻,Hibernate 或 OJB 会根据存储逻辑自动生成 SQL 并调用 JDBC 来加以执行。

2.2　半自动对象关系映射的特点

半自动对象关系映射的实现如 Apache iBatis 等,它不同于 Hibernate 这样的全自动对象关系映射对数据库进行全自动化的封装。它主要是定义对象到 SQL之间的映射关系,在运行时刻它不会自动生成 SQL,具体的 SQL 需要由程序员来编写,然后通过映射文件,将 SQL 所需要的参数,以及返回的结果字段映射到对象。

2.3　纯 ORM 与半自动 ORM 的优缺点

与全自动对象关系映射相比,半自动对象关系映射具有如下优点:

1)开发人员可以对数据库操作细节进行完全的控制,在数据处理量大、对性能要求比较苛刻的情况下,比较有利于 DBA 对 SQL 语句进行优化和调试。

2)能够更好地适应面向对象模型与数据模型设计不好的系统,以及遗留系统、第三方数据库和对现有数据库的复用。

3)很容易实现动态 SQL,极大地方便了开发。

4)系统使用简单,降低开发人员学习负担。

半自动对象关系映射所带来的缺点:

a. 比较而言,加大了开发工作量。

b. 如果在 SQL 语句中使用了某一数据库特有的特性,则会对可移植性造成一定影响。但是,在全自动对象关系映射中,根本就拒绝了开发人员对数据库私有特性的访问,而半自动 ORM 则由于其开放式的设计,为开发人员提供了访问数据库各种特性的途径。

应该注意到,并不存在着全自动 ORM 与半自动 ORM 哪一个更好的问题,它们都有着各自不同的适用场合。对于新系统的开发,封闭化的设计能够带来更好的开发效率和更好的封装机制,但是在某些情况下,却又为一些必需的底层调整带来了阻力,如在对遗留系统的改造和对现有数据库的复用上,表现出灵活性不足的弱点。在这种情况下,半自动对象关系映射的表现更好[3]。

3　系统设计与实现

3.1　设计思想

本系统对 JDBC 进行轻量级的封装,系统所提供的接口最终会被转换成 JDBC 调用。由于是对半自动对象关系映射的实现,因此系统并不会为程序员在运行期自动生成 SQL 执行,具体的 SQL 需要由程序员编写。

SQL 语句(实际上是 PreparedStatement 语句,或存储过程)由程序员在 XML 映射文件中进行编写,同时在 XML 文件中定义参数映射信息以及结果集映射信息。

其中参数映射信息的作用是将作为参数的 Java 对象的属性映射到 PreparedStatement 语句的参数并为它赋值;结果集映射信息将查询结果集 ResultSet 的字段映射到作为结果返回的 Java 对象的属性并为它赋值。这样一条记录将构造一个 Java 对象,如果返回多条记录,则可以使用 List 来存储所构造的多个对象。由此可以利用 Java 对象来返回查询结果。

3.2　系统体系结构

图 2 为持久层的体系结构,主要由五部分构成:映射生成器、映射规则库、对象存取器、缓存管理器、事务管理器。

3.2.1　系统外部实体

与系统交互的外部实体主要有:

1)商业逻辑类。商业逻辑类使用持久层提供的服务,它可以向持久层请求事务,发出查询、插入、更新、删除记录等请求;

2)参数对象。参数对象主要有两种,简单类型如 Integer,String,当需要传递到 SQL 的参数只有一个时就用简单对象;但是经常会遇到需要传递多个参数到

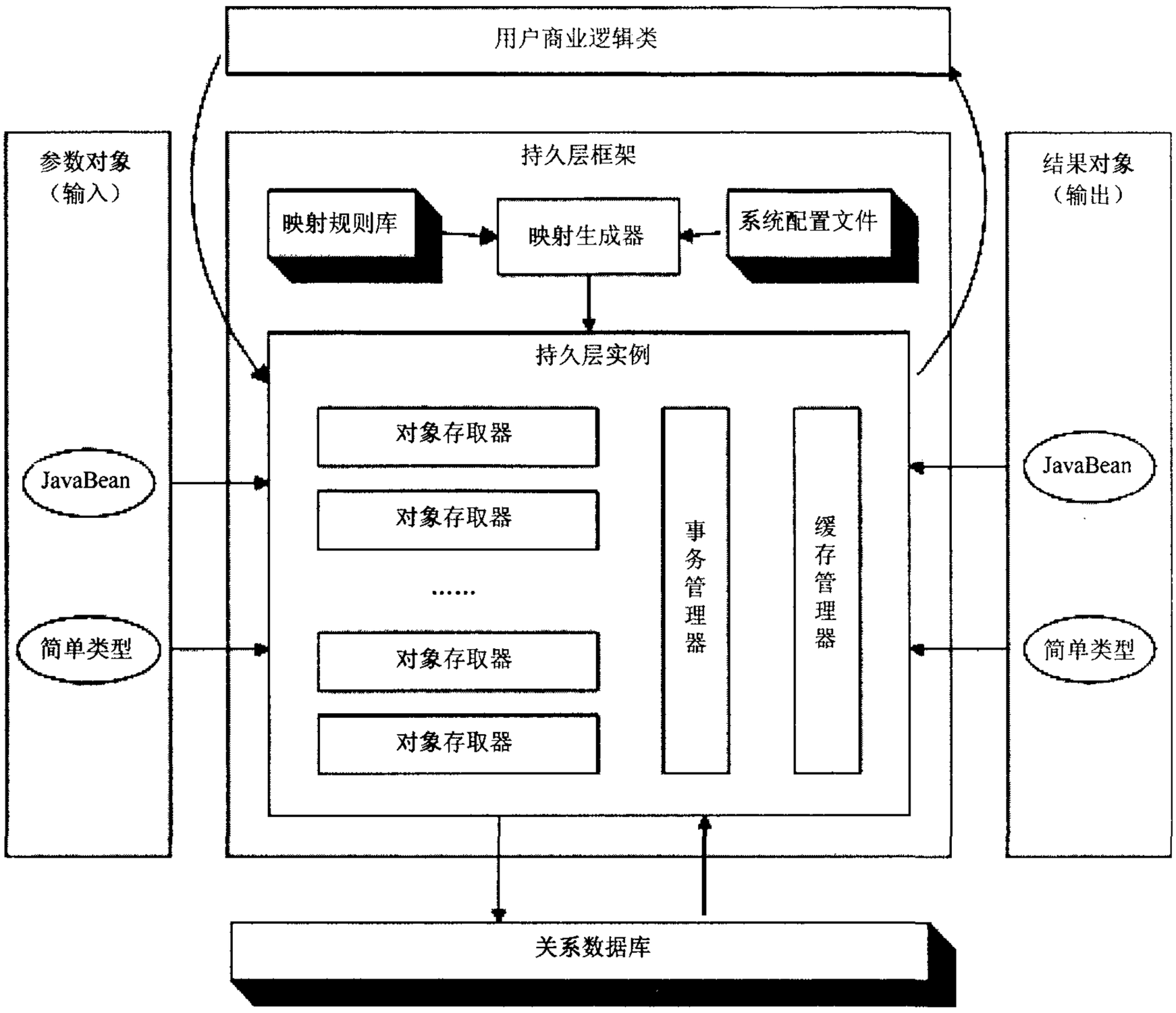

图 2 系统体系结构

SQL 的情况，这时需要使用一个 JavaBean 对象来封装多个参数。

3)结果对象。从数据库中查询到的结果集经过结果集映射处理之后，将被转换成结果对象，然后由持久层返回给商业逻辑类来进行处理。

4)关系数据库。

3.2.2 配置解析

它主要包含三部分，分别是：系统配置文件、映射规则库和映射生成器。

(1)系统配置文件提供了对系统的各种属性及行为进行定制的手段，例如数据库连接机制，是否启用缓存机制或延迟加载机制等，另外它还指定了映射规则库的位置信息；

(2)映射规则库中封装了 Java 对象到关系数据库的映射信息。主要包含三部分，分别是 SQL 语句，参数映射信息，结果集映射信息。

(3)映射生成器读取系统配置文件以及映射规则库中的信息并进行解析，然后利用解析到的信息构造一个持久层实例。

3.2.3 持久层实例

持久层作为系统核心组件向应用程序开发人员提供各种服务，如对事务的支持，对查询以及更新操作的处理等，用户与数据库的所有交互都通过这个实例来完成，它包括如下三个部分：

(1)对象存取器。

对象存取器是系统中的重要组成部分，映射生成器在对映射规则库进行解析之后，将会在系统中生成若干对象存取器。它实际负责与数据库的交互，持久层实例所接受到的用户的对数据库的交互请求，如查询、插入、更新、删除等，将被转交到对象存取器来执行。它主要包含三个部分，如图 3 所示。

各个部分作用如下：

①参数映射实体将参数对象的属性映射成 SQL 语句所需参数，并为 SQL 参数赋值；

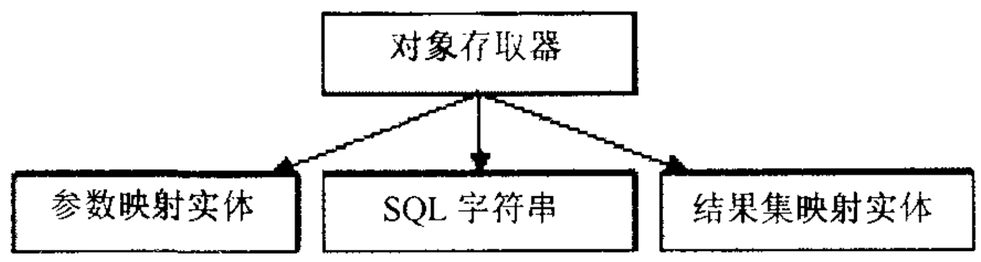

图 3 对象存取器

②结果集映射实体将查询结果集 ResultSet 的字段映射到结果对象的属性并为之赋值，以此构造结果对象；

③SQL 字符串是一个 PreparedStatement 语句或者存储过程，所需要的参数的值从参数映射实体中获取。

在运行时，一个对象存取器分别维护一个参数映射实体、结果集映射实体，以及 SQL 字符串。对象存取器利用参数映射实体来为 SQL 字符串的参数赋值，然后执行 SQL：

* 如果是更新 SQL，则返回受影响的记录数；

* 如果是查询 SQL，则利用结果集映射实体将查询结果集转换成对象，并返回给持久层实例。

(2)缓存管理器。

缓存管理器为持久层提供对对象进行缓存的功能。当在应用程序中进行数据库查询操作时，首先根据所使用的查询语句的特征构造缓存的主键，然后以

（下转第 94 页）

是描述空间关系学说的理论;而粗糙集理论主要是研究粒度的表示,刻画和粒度与概念之间的依存关系。更主要的不同在于:商空间理论是在论域元素之间存在有拓扑关系的情况下进行研究的,即论域是一个拓扑空间,而现在的粗糙集理论其论域只是简单的点集,元素之间没有拓扑关系。从这个角度说,粗糙集只是商空间理论的一个特例,即不同粒度均在同一种空间结构中进行,没有粒度空间的变化[9]。另外,粗糙集是在给定的知识基上求解对应的问题,如求集合的 R^- 上近似和 R_- 下近似,我们是在(X,T)中讨论各商空间之间的关系,求相应的(各种意义下)上近似空间和下近似空间。从这个角度看,可以说粗糙集是微观的粒度计算,商空间理论是宏观的粒度计算。这两个理论都是建立在等价关系之上,所以可以将两者结合起来。

与商空间理论相比,词计算理论主要是讨论粒度的表示问题,即当人类进行各种思考和推理时,都离不了粒度,于是用"语言"、"词(word)"来表示,进而牵涉到"词计算"问题的"模糊数学"方法,而利用模糊等价关系可以将原来的商空间理论推广成模糊商空间理论,故两者具有等价性[10]。

4 结束语

以上简单介绍了商空间理论、词计算理论、粗糙集等粒度计算方法之间的关系。可以看出这三个不同的粒度计算理论,从思考问题的出发点和解决问题的任务,都不尽相同,各有千秋。但是三者都有一个共同的特点,那就是都考虑到人类智能中,有从不同粒度思考问题的这一特点。相信将商空间理论与粗糙集理论以及模糊数学方法相结合,取长补短,将能给出一个更强有力的粒度计算理论和方法。

参考文献:

[1] Zadeh L A. Fuzzy logic = computing with words[J]. IEEE Transactions on Fuzzy Systems, 1996, 4: 103 - 111.

[2] Zadeh L A. Towards a theory of fuzzy information granulation and its centrality in human reasoning and fuzzy logic[J]. Fuzzy Sets and Systems, 1997, 19: 111 - 127.

[3] Zadeh L A. Announcement of GrC[EB/OL]. 1997. http://www.cs.uregina.ca/yyao/GrC/.

[4] 李道国,苗夺谦,张红云:粒度计算的理论、模型与方法[J].复旦学报,2004(5):838 - 841.

[5] Pawlak Z. Rough Sets[J]. International Journal of Computer and Information Science, 1982, 11: 341 - 356.

[6] 张 媛,张 铃,张燕平.粗糙集算法及其应用[J].微机发展,2005.15(4):17 - 18.

[7] 许中卫,李龙澍.基于粗糙集理论的数据挖掘算法研究[J].微机发展,2001,11(1):6 - 9.

[8] 张 钹,张 铃.问题求解的理论及应用[M].北京:清华大学出版社,1990.

[9] 张燕平,张 铃,夏 莹.商空间理论与粗糙集的比较[J].微机发展,2004,14(10):21 - 24.

[10] 张 铃,张 钹.模糊商空间理论(模糊粒度计算方法)[J].软件学报,2003,14(4):770 - 776.

(上接第 90 页)

该主键在缓存中查找,如果存在对应记录,则直接返回该对象;如果不存在,则在数据库中进行查找,并将查找结果存放于缓存中,然后将查找结果返回给应用程序。在实际应用中,对于不同的对象可能需要不同的缓存形式,因此,本系统将缓存实现为一个可配置的形式,在使用时可以根据需要灵活地选择不同的缓存。

(3)事务管理器。

事务管理器为持久层提供事务管理的功能,持久层实例将从事务管理器中取得数据库连接。本系统并不打算独立实现事务管理,而是将它委托给底层事务管理实体,如 JDBC 事务或者 JTA 事务。

4 结束语

文中对对象关系映射进行了简要的介绍,并分析了半自动对象关系映射和全自动对象关系映射的特点,给出了半自动对象关系映射的一个设计。该持久层具有一定的对象关系映射功能,使应用程序开发人员能够以一种统一的、面向对象的方法进行对象的存取,降低开发负担,同时有效地分离了业务逻辑与数据库访问逻辑,有利于应用系统的可维护性和可扩展性。

参考文献:

[1] 孙卫琴.精通 Hibernate:Java 对象持久化技术详解[M].北京:电子工业出版社,2005.

[2] Ambler S W. The Design of a Robust Persistence Layer for Relational Databases [EB/OL]. 2005. http://www.ambysoft.com/downloads/persistenceLayer.

[3] 夏 昕,曹晓钢,唐 勇.深入浅出 Hibernate[M].北京:电子工业出版社,2005:468 - 469.

[4] Ambler S W. Mapping objects to relational databases What you need to know and why[EB/QL]. 2000. http://www-128.ibm.com/developerworks/webservices/library/ws-mapping-to-rdb/.

[5] Ambler S W. Mapping Objects to Relational Databases: O/R Mapping In Detail[EB/QL]. 2000. http://www.agiledata.org/essays/mappingObjects.html.