

Java 中的持久性存储技术——JDBC 和 JDO

冯 战,郝克刚,葛 玮

(西北大学 软件工程研究所,陕西 西安 710069)

摘 要:在实际的企业应用程序中,开发人员可以通过多种方式来实现数据的持久化存储。例如,使用 Java 序列化并存储整个对象的二进制表示;较通用的企业方案是通过 EJB 来使用应用程序服务器的工具;或者使用 JDBC 存储在数据库中,还有就是通过 JDO。文中主要介绍了 JDBC 和 JDO 技术,并把 JDBC 和 JDO 作了比较,讨论了各自的特性和不同点,并突出了常见的误解。最后得出一般性的结论:它们是互补的 API,每个都为各自的目的而很好地服务。

关键词:JDO;JDBC;持久性存储;序列化;域对象模型

中图分类号:TP311

文献标识码:A

文章编号:1673-629X(2006)09-0046-04

The Persistence Store Technique in Java — JDBC and JDO

FENG Zhan, HAO Ke-gang, GE Wei

(Software Engineering Institute, Northwest University, Xi'an 710069, China)

Abstract: In practice of the enterprise application, developer can implement data's persistence store with many ways, for example, using Java serialization and storing the binary presentation about the whole object; more popular enterprise scheme is to use the tools of application server through EJB; storing in the database through JDBC or JDO. Introduces the technique of JDBC and JDO, compares JDBC with JDO, discusses their characteristics and differentia, and shows usual misunderstanding. Final conclude in a general way: they are the complementary API, and every one serves well with their different intents.

Key words: JDO; JDBC; persistence store; serialization; domain object model

0 引 言

对于持久性的需求是任何企业应用程序的先决条件,在使用 Java 语言编写应用程序时也是一样的。许多 Java 应用程序都是有将应用程序数据保存在某种存储设备并从中取出而使用的代码组成的,以便以后能够检索或使用它们。

Java 是一种面向对象语言,开发人员本能地使用对象工作,对象在任何时候都由其成员字段表示状态。从程序员的观点来看,持久性关注获得的是应用程序特有的信息,这些信息事实上是对象的状态,然后使这些信息在超出虚拟机生命周期之后仍然可用。简而言之,即使 Java 虚拟机关闭后,数据也应该可用。

开发人员可以通过多种方法来实现这一想法:或者使用 JDBC 存储在数据库中;使用 Java 序列化并存储整个对象的二进制表示;较通用的企业方案是通过 EJB 来使用应用程序服务器的工具。不考虑技术上的因素以及涉及到的

复杂程度,这些方法都有一个共同的概念特征:当需要存储状态时,由开发人员从对象中获得状态信息,当应用程序需要时再重新建立这种状态,还有就是通过 JDO。文中主要介绍了两种技术:JDBC 和 JDO。

1 JDBC

在 Java 应用中,JDBC 提供了一个接口用来发布 SQL 命令,是目前运用最为广泛的标准 API。它使用 Java 语言来访问关系数据库,但是 JDBC 要求明确地处理数据字段,并且将它们映射到关系数据库的表中。将底层数据仓库中的信息映射到其程序表示中是开发人员的任务,如图 1 所示。

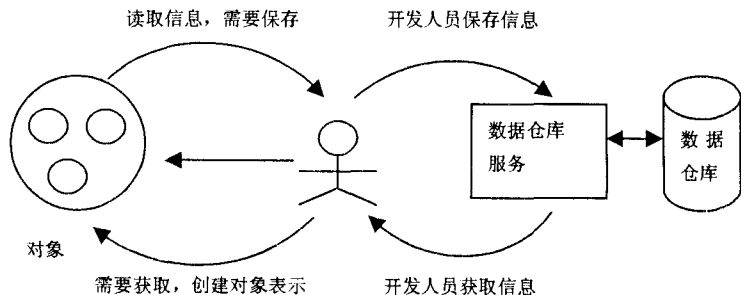


图 1 JDBC 处理信息的持久性

JDBC 使用 SQL 的关系数据模型作为它的数据表示,它处理的粒度是基于二维表格的一个单元格。不同的关

收稿日期:2006-01-08

作者简介:冯 战(1981-),男,陕西西安人,硕士研究生,研究方向为中间件技术、软件工程;郝克刚,教授,博士生导师,研究方向为软件工程与相关理论、工作流、中间件;葛 玮,副教授,硕士生导师,研究方向为软件工程与相关理论、工作流、中间件。

系数据库采用的 SQL 语句在其它的 RDBMS 中不一定可以运行,严格讲 SQL 语句缺乏不同数据库之间的移植性^[1]。

开发人员被迫与两种区别非常大的数据模型、语言和数据访问手段打交道:Java 对象模型,以及 SQL 中的关系数据模型^[2]。并且说明两者之间的关系。它们还要保证两个模型的视图及行为保持一致,因为对二者之一任何细小的改变都将导致模型与现实世界业务之间的不匹配。如图 2 所示。

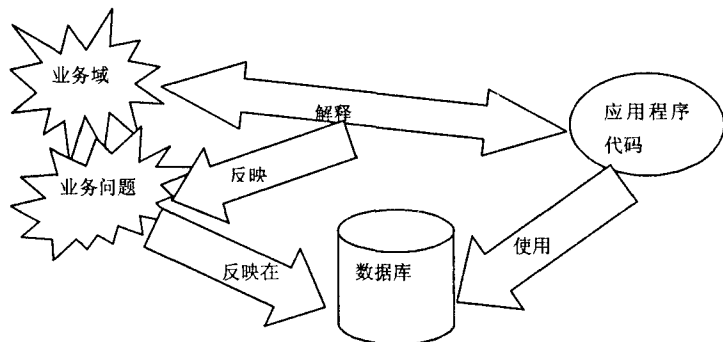


图2 JDBC的持久性导致的两个模型

虽然 JDBC API 明显是面向对象的,但是 SQL 实际上是过程化的,JDBC API 仍然是 SQL 的包装器,JDBC 中持久数据总是以行或者列的形式出现,从来不会直接以 Java 对象的形式出现,而且由于 JDBC 中的 SQL 处理的数据模型是二维表格,不是 Java 类,这样就给编程人员带来了一些问题,任何面向对象的思想都要求采用对象的处理方式建模数据,而 JDBC 又不能处理对象。自然基于 JDBC 的方式采取一些过程化的编程逻辑,而较少的编程人员既熟悉 Java 开发又熟练掌握 SQL 语句。结果代码不好维护、重用性低,低效的 SQL 语句大量的消耗资源,完全丧失了面向对象的优点。

2 JDO

在 Java 应用程序中,JDO 能够实现 Java 对象的直接存取并且支持事务、查询和对象缓存的管理。在 JDO 中数据源对于 Java 程序来说是透明的,无论是关系数据库(RDBMS),还是面向对象数据库(OODBMS),甚至是文件系统(FS),Java 程序都可不加修改地运行^[3]。

JDO 将同时满足两种目标:一个目标是提供应用程序代码和低层持久性仓库(如数据库以及类似于 JDBC 的文件系统)之间的接口。另一个目标是通过提供可以操作并跨越持久对象的 Java 中心机制来简化安全的、可伸缩的应用程序开发。JDO 在这两方面都做的很好。

JDO 独到的核心思想是在尽可能不增加程序员额外工作的情况下提供一个面向 Java 对象的数据库存储机制^[4]。对 Java 开发人员而言,JDO 为对象持久性提供了

第一个标准化的、完全面向对象的方法。如图 3 所示,如

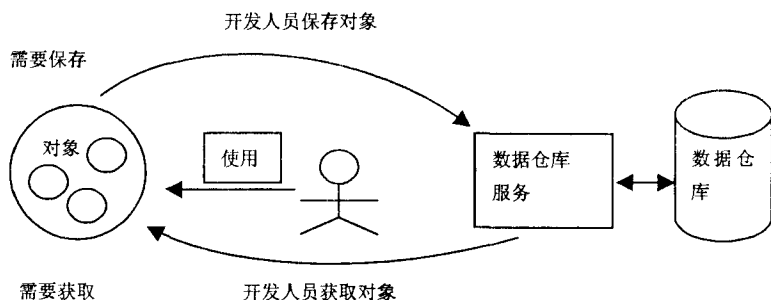


图3 JDO处理信息的持久性

何持久化并重新建立对象的细节对开发人员来说是透明的;这经常称为透明的持久性。JDO 的透明性意味着不再需要明显的数据库调用,而且开发人员不用担心对象和数据库之间映射的基本代码。在 JDO 的应用中,应用程序不需要在数据对象之间提供任何映射,它也不需要去跟踪什么被改变了并且把这些变更重新映射到底层数据仓库的数据对象中。应用程序只需要去遍历它在内存中的对象模型,在运行时刻 JDO 只关心底层数据仓库所提供的实例。

用透明的持久性进行设计将非常简洁。开发人员只需要编写并维护一个模型或程序的表示。当需要的时候,可以使用透明的持久性服务来持久化模型的实体。通过 JDO,开发人员可以集中精力围绕主要的目标来设计应用程序,而不是围绕低层的持久性机制来设计应用程序。如图 4 所示。

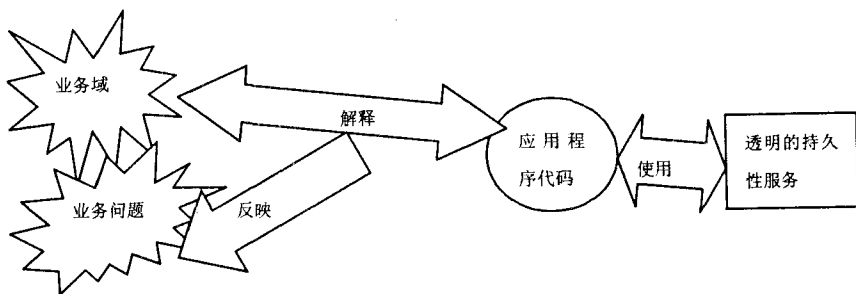


图4 JDO中透明的持久性模型

JDO 提供了一个二进制的兼容接口用来存储 Java 对象^[5],是独立于任何数据源的,它尝试抽象出实际的数据源。这允许不同类型的数据源“插入”到已部署的系统中。现在可以使用对象数据库或甚至是驻留在文件系统上的 XML 文件作为数据源,而不是被限定在后端的关系数据库上^[6],使得应用可移植性很强,减少了开发和部署的成本。还有,JDO 实现可以即插即用^[7]。在使用 JDO 的应用程序中,JDO 实现作为独立的数据持久层存在,因而是可插拔的,不同数据源的 JDO 实现能够被方便地替换,同时它们也很容易被集成到应用服务器环境中去。

3 关于 JDBC 和 JDO 的一些错误理解

随着 JDO 的出现,开发人员会问 JDO 怎样和现有的

数据访问技术(如 JDBC)相关联。必须明白 JDO 采用了和 JDBC 不同的方法来进行持久化,所以经常会产生误解。

1) JDO 并不是 JDBC 的替代品,已经存在的使用 JDBC 来完成的应用程序仍然可以继续使用,而且可以认为 JDO 是 JDBC 的一项补充技术。

首先,只要你处于 O/R 映射情况中(如果你想要使用关系数据库作为基于 JDO 的应用程序的最终数据仓库的话),那么在此情况下,JDO 并没有真正代替 JDBC;相反,它把应用程序开发人员和它的细节相隔离,并且允许开发人员在较高层次上编写代码。然而,JDO 实现仍然需要在其内部使用 JDBC 和一个 JDBC 驱动程序,以此来访问位于较低层次上的关系数据库。当然,如果你使用的是一个非关系数据库的话,情况就将有所不同。

其次,对于更复杂的企业应用程序,有时候需要带有特殊的情形(取决于一些 JDBC)的 JDO 的混合。你可能会把这些 PreparedStatements 和 ResultSets 用于自身应用程序的特定模块中,以便在后面使用。最后,对 JDBC 有一个很好的了解,这对于解决基于 O/R 的 JDO 实现的各种问题确实很有帮助。

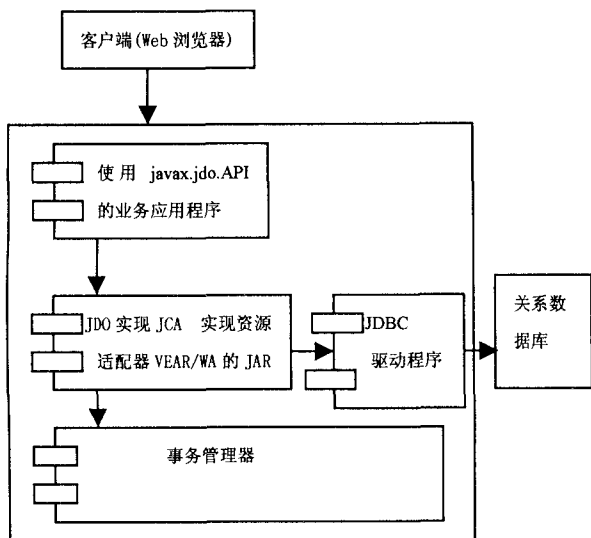


图 5 在基于 JDBC 的关系数据库中 JDO 的使用

图 5 给出了一个典型的系统体系结构,该体系结构给出了在应用程序服务器内部运行的解决方案。在应用程序服务器端,某些业务逻辑使用 JDO API,该 API 又在内部使用 JDBC API。

2) 性能。

性能是开发人员经常关注的一个重要问题。在实际情况中,可能有的人会认为“JDBC 比 JDO 快”,但情况并不总是这样。基于 JDO 的应用程序和基于 JDBC 的应用程序这两者的性能在很大方面取决于实际应用程序的情况、特定的用例以及如何使用 JDBC 的细节等。而且,在实际情况中会发现其实在很多情况下基于 JDO 的应用程序可能比基于 JDBC 的应用程序快,其根本的原因在于它避免了对数据仓库的远程往返,这将会节省很大的开销。

JDBC 并没有内建的性能优化,开发人员必须明确设计 JDBC 的性能优化。在 JDBC 中构建客户端缓存和按请求获取等重要的性能优化,开发代价是非常大的。JDBC 的预备查询语句是非常有用的性能优化。如果 RDBMS 支持预备查询,频繁的查询可以被实例化一次后反复地执行。JDO 的性能优化很一般,它明确使用客户端缓存的编程模型。这个缓存由 JDO 实现来管理,而无需应用程序的介入。但是高级的应用程序仍然可以在程序上控制客户端的缓存,因为 JDO 提供了这些方法,例如回收缓存对象或从数据存储中刷新状态。

3) 数据库的独立性。

虽然 JDBC 驱动程序的类概括了产品特有的客户端/服务器连接协议细节,但是 JDBC 并没有试图“翻译”SQL 代码。然而 JDO 实现根据低层数据仓库翻译独立于数据仓库的 JDO 查询语言(JDOSQL),可以考虑到 SQL 的细微差异并进行相应的映射。

由于以上的原因,通过 JDO 使用来自于同一项目的不同关系数据库比通过 JDBC 更为容易,JDO 可能隐藏了数据库供应商特有的 SQL。如果将来的目标是非关系数据仓库,如对象数据库,那么 JDBC 无论如何都不能作为持久 API 的选择,因为 JDBC 主要是面向关系数据库的。

4 对两种持久性存储技术的选择

4.1 何时在应用程序中使用 JDBC

在实际的情况中,一些应用程序可能需要实现用例,这时还是要用到 JDBC。因为这些常常不是普通的示例应用,而是一些复杂的解决方案的媒介。概括地说,这种情形一般是下面几种情形中的一种:

1) 特定的 SQL 查询不能在 JDOSQL 中表示。JDO 标准查询语言(JDOSQL)并不是为了试图取代 SQL 的,它的目标不是能够表示任何人们可能想到的 SQL 查询。

2) 需要使用现有的、通常是供应商特有的、关系数据仓库的特性。关于特定关系数据仓库的特性的使用,通常是供应商专有的 SQL 扩展或者整个应用程序。

3) 以数据为中心的、处理密集的、性能敏感的(批量的)应用程序。JDO 主要用于以域对象模型为中心的应用程序,但是其他应用程序也更多地使用 JDBC 而不是 JDO。例如,主要以数据为中心的处理过程可能需要减少使用现有关系数据库特性的处理。

4) 现有的关系数据库模式可能很复杂,并且不能被更改。有时候我们会要求一些应用程序在现有的传统关系数据库模式下工作。如果这种关系数据库模式实际上是复杂的关系数据库“应用程序”,例如,使用 SQL 视图和存储过程来访问数据,则原始的 JDBC 可能比 JDO 更加合适。

总之,在实际的开发过程中,对于以数据为中心的应用程序,它的主要目的是保存和检索面向记录的数据,JDBC 是最适合开发这类应用程序的工具。面向记录的数据

可以很容易地直接映射到这种关系模型,并且应用程序能够权衡 SQL 针对特殊查询和数据访问的能力。这里重点是数据。

4.2 何时在应用程序中使用 JDO

对于以对象为中心的应用程序,它的主要目的是处理相互关联的数据的复杂图或者层次结构,JDO是最适合开发这类应用程序的工具。对象模型可以用于表示相互关联的复杂图或者层次结构,这里应用程序可以通过访问这些数据来完成处理。此处的重点是对对象模型。

4.3 何时在应用程序中使用 JDO 和 JDBC

有时候,你可能希望把 JDO 代码和一些 JDBC 结合起来使用。虽然应该避免这种混合的体系结构(除非有很好的理由使用它,而且这种体系结构严重限制了其他数据仓库的可移植性),但它在技术上是可行的。一个方法是使用实现特有的 API 扩展来使用 SQL 而不是 JDO SQL(或者两者混合使用)作为查询语句。另一个方法是根据应用程序需要使用混合体系结构的原因来使用 JDO 实现特有的方法获得 Connection。Connection 应该通常来自于某个 JDO 的 PersistenceManager。

5 结束语

JDO 是纯 Java 的 API^[8],并且是为透明对象的持久性而设计的高级 API。程序员通常会编写 Java 代码,并像往常一样通过 Java 对象进行工作。对象中的数据总是被自动持久化,可以在应用程序的不同运行中重复使用,可被查询,甚至其他应用程序也可以访问这些数据。所有这些都可以通过一个相对透明的方式来实现^[9]。它综合了以前所有持久存储 APIs 的最好的特征,而没有相关的缺陷,它使得应用程序开发人员可以做到“只写一次,处处持久”。

JDBC 是专门为访问关系数据库而设计的低级 API,因为在任何地方都不可能把关系数据库以任何方式“隐藏”起来^[9],而且它通过一个基于 SQL 的接口提供了可靠性和可测量性。JDBC 被应用于关系模型而使得它很难和

应用程序中的对象模型相结合,这是因为 JDBC 不支持 Java 类,所以 JDBC 必须直接和 SQL 的数据模型一同工作。

通过 JDBC 你可以具体说明哪些内容应该被持久化和这一切是怎么发生的。通过 JDO 你仅仅定义了你的类中的持久部分,至于是怎么被持久化的取决于 JDO 的供应商。两个 API 显然是根据编程者心中的不同目标而设计的,虽然两个 API 都是要在高层次上为“持久数据”服务的,但实际上,它们只是通过不同的方法来达到这个目的。而且它们是互补的 API,两者都具有独特的功能,可以被不同层次的程序员使用,并可以根据不同的开发目的而应用到项目或项目的模块中。

参考文献:

- [1] 刘柯,杨贯中. J2EE 工程中持久性存储技术的比较[J]. 株洲师范高等专科学校学报,2004,2(9):43-45.
- [2] Jordan D, Russell C. Java Data Objects[M]. USA: O'Reilly, 2003. 1-2.
- [3] 何成万,余秋惠. JDO 初探[J]. 计算机工程,2002,28(6):282-283.
- [4] 翟鸿鸣,张惠娟. JDO 技术研究[J]. 微机发展,2004,14(6):78-81.
- [5] Jordan D. A Comparison Between JDO, Serialization and JDBC for Java Persistence[EB/OL]. <http://www.jdocentral.com/pdf/DavidJordan-JDOversion-12Mar02pdf>, 2002-01.
- [6] 张伟燕,夏涛,席传裕. 在 Java 企业应用中选择正确的对象持久技术[A]. 中国计算机科学与技术-2004. 合肥:中国科学技术大学出版社,2004. 945-946.
- [7] 夏科军,徐良贤. JDO 实例的状态管理研究[J]. 计算机仿真,2005,22(4):269-272.
- [8] Jones B L. Data Persistence and Java Data Objects—JDO[EB/OL]. <http://www.developer.com/java/article.php/918111>, 2002.
- [9] Tyagi S, Vorburger M, McCammon K, et al. JDO 核心技术[M]. 北京:清华大学出版社,2005. 258-259.

(上接第45页)

润情况、客户群分布等重要信息。并结合大盘走势,提供不同行情条件下的最大收益经营方式。同时,通过对各营业部经营情况的横向比较,以及对本营业部历史数据的纵向比较,对营业部的经营状况作出分析,提出经营建议。通过对客户状态、交易行为、自然属性和其他信息的综合分析,细分客户群,确定核心客户。同时通过对其进行关联分析,可为协助制定各种有效的营销方案,开展针对性的个性化服务。今后还应继续采用预测、相关、关联等技术,挖掘更深层次的规律^[6]。

参考文献:

- [1] Berson A. 构建面向 CRM 的数据挖掘应用[M]. 北京:人

民邮电出版社, 2001.

- [2] Hand D, 张银奎. 数据挖掘原理[M]. 北京:机械工业出版社, 2003.
- [3] 陈京民. 数据仓库与数据挖掘技术[M]. 北京:电子工业出版社, 2002.
- [4] Han Jiawei, Kamber M. Data Mining Concepts and Techniques(影印版)[M]. 北京:高教出版社, 2001.
- [5] 邹涛,戚广智,蔡丽娟,等. 网络信息挖掘系统 IIGS 的实现[J]. 南京大学学报(自然科学版), 2000(2):183-188.
- [6] 胡乐群. 数据挖掘在证券行业中的应用[EB/OL]. <http://industry.ccidnet.com/art/465/20020904/24238-1.html>, 2002-09-04.