第 33 卷 第 5 期 计 算 机 工 程 2007 年 3 月
Vol.33 No.5 Computer Engineering March 2007
·开发研究与设计技术· 文章编号：1000—3428(2007)05—0272—03 文献标识码：A 中图分类号：TP391.73

基于 J2EE 的数据持久化的研究与实现

苗晓辉
（浙江大学计算机科学与技术学院，杭州 310027）

摘 要：数据持久化是 J2EE 平台开发过程中一个重要组成部分，也是构建企业级应用系统面临的棘手问题。该文摒弃基于 J2EE 构建企业级应用系统过程中典型的数据持久化技术，提出了通过使用 EJB 中的 SessionBean，在适度使用配置文件的基础上构建规范的数据持久化平台，实现系统数据调用与应用逻辑分离，保证了数据调用线程安全，为系统的扩展、移植、重组维护打下了一个良好的基础。
关键词：企业应用；J2EE；持久化；数据集成

Design and Implementation of Data Persistent Based on J2EE

MIAO Xiaohui
(College of Computer Science & Technology, Zhejiang University, Hangzhou 310027)

【Abstract】 Data persistent is an important part of development in J2EE, and is an intractable problem in the process of designing enterprise application system. Discarding the type technology of data persistent in J2EE, it uses SessionBean technology of J2EE and configuration files within measure, forms normative data persistent platform, implements the separation of data transfer and application logic, assures the thread safety of data transfer, in the same time, laies the foundations of system extension, transplant, recomposition.
【Key words】 Enterprise application; J2EE; Persistent; Data integration

面向企业、团队协同合作是目前企业级应用系统开发的主流模式，在这种开发模式下系统各模块之间、系统与底层数据库之间的数据交互很频繁，开发过程中数据调用与业务逻辑混在一起极易分散业务逻辑开发人员精力，同时也为系统更新、模块复用、功能重组复杂化埋下伏笔。因此有必要建立一个规范的数据持久化层，以封装系统中的数据调用逻辑，通过提供统一的数据调用接口，简化开发人员数据调用代码复杂度，减少系统更新模块重组时不必要的代码重写。

本文所述数据持久化技术基于 J2EE 构架下 SessionEJB 实现，直接应用对象是浙江省“杭州市重大关键技术项目”中的“基于 XML 技术的棉纺企业信息系统(棉纺织综合信息管理系统)”。本文所述数据持久化技术可作为构造企业级应用系统时通用数据持久化技术使用，也可作为开发电子商务或电子政务时的数据交互技术。

1 数据持久化技术概述

1.1 持久化的概念

持久化指对象的生存特性，如果对象的生存期跨越程序执行期，则称此对象具有持久化性质。数据是对象的一种，具有持久性的数据称为持久性数据(persistent data)。持久性数据改变了数据的生存状态，暂态数据典型生存周期是：被创建→被使用→被删除，数据的内部状态随着数据生存周期结束而消失，而持久性数据则通过保存到持久存储区(如文件、数据库等)的方法延长数据持久生存状态。持久性数据被删除之前，已经将自己的状态持久化。这样，持久性数据不需要创建，通过持久化存储和检索，就可以将数据恢复使用。目前，实现数据持久化的方法主要是映射策略，通过建立持久性数据到数据源的映射实现数据持久化，即将各持久性数据与数据库中的表、字段通过配置文件建立一一对应的关系，各持久性数据知道自己需要被保存到数据库中哪个表、哪个字段。

1.2 典型的数据持久化策略

基于映射策略的数据持久化使系统应用逻辑与数据存储分离，有利于系统的管理、维护和扩展，J2EE 技术下数据持久化策略有诸多较成熟的框架和产品，其中有代表性的主要是：

(1)实体 Bean。实体 Bean 是 J2EE 中 EJB 的一种，主要通过映射文件实现数据持久化。就一张数据源的表而言，各字段被映射到实体 Bean 中相应属性；表与表之间的关系，相应地被映射成不同实体 Bean 之间的关系。

(2)Java 数据对象。Java 数据对象(Java Data Object, JDO)也是数据持久化策略的一种，它是一个存储 Java 对象的规范，已经被 JCP 组织定义成 JSR12 规范，规范的两个主要目的是提供数据处理和访问机制的 API，并允许依此规范的实现作为应用服务器一部分。JDO 在实现数据持久化的过程中，通常会有一个由 JDO 实现厂商提供的工具来完成普通 Java 类到 JDO 实例的转化，在此过程中，JDO 工具在适当时候会修改 Java 类字节码，将普通 Java 类改造成 JDO 实例，以达到数据持久化的目的。

(3)Hibernate。Hibernate 是按照 LGPL 许可证发布的开放源代码应用程序，主要提供 Java 超高性能的对象/关系持久性和查询服务。Hibernate 是一个优秀的开放源代码数据持久层轻量级封装框架，不需要任何容器，不过也可以整合到 J2EE 系统中作为数据持久层框架。Hibernate 在 Java 应用程序中不

基金项目：国家自然科学基金资助项目(60502027)；深圳市科技计划基金资助项目(200336)
作者简介：苗晓辉(1975－)，男，硕士生，主研方向：信息集成与数据安全
收稿日期：2006-03-11 **E-mail：**xiaohuimiao@sohu.com

用作任何修改就可以直接映射 JavaBean，从而取代大部分 JDBC 代码。Hibernate 通过提供简单易用并符合 ODMG3－style 的 API，解决了 JDO 和实体 Bean 的一些缺陷。

1.3 现有 J2EE 技术下数据持久化策略的不足之处

J2EE 技术框架下各数据持久化策略均是基于映射策略，并无一例外地采用基于大量配置文件的方式实现数据持久化。这些配置文件给开发人员、配置人员和系统维护人员带来极大不便，对开发人员而言，不但要维护持久数据的代码，同时要维护与持久数据对应的配置文件信息。当然，多数 J2EE 集成开发环境提供配置文件自动生成工具，但也是在开发人员熟知配置文件信息基础之上才能如此；对系统配置人员而言，将拥有大量配置文件的系统各模块准确无误地进行装配，简直是一场噩梦；同时，由于系统信息离散分布于代码和配置文件之中，因此也增加了系统维护、移植、重组和集成的难度。上述基于配置文件的数据持久化策略，只是降低了系统开发人员首次开发工作难度，而将系统的移植、重组、集成和配置工作转嫁于相应的配置、维护人员，并没有从根本上解决问题。

针对以上问题，在开发“棉纺织综合信息管理系统”过程中，我们采取适度使用配置文件的策略，借助于 EJB 中会话 Bean 技术，通过只将数据持久化的必要信息放在配置文件的方法，减轻系统装配、维护人员工作难度，同时也增强了程序可维护、重组、更新性能。

2 数据持久化层的总体结构设计

2.1 数据持久化层在系统中的位置

本文所述数据持久化技术直接应用对象“棉纺织综合信息管理系统”是基于 J2EE 架构的分布式系统，采用 B/S 结构进行设计；UI 层由 JSP 和 Servlet 组成，负责生成用户界面，是与用户交互的窗口；业务逻辑层由 J2EE 容器的业务对象负责具体业务逻辑；数据持久化层位于业务逻辑层之下，也是 J2EE 容器的对象，但其专司系统与数据库间的数据交互。数据持久化层在系统中的位置如图 1 所示。

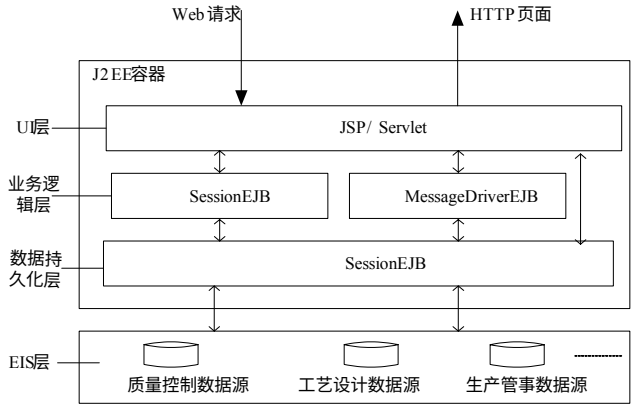

图 1 数据持久化层在系统中的位置

数据持久化层的主要功能是实现数据存取操作与业务逻辑的分离，通过在业务逻辑对象与数据源之间提供一个简洁规范的调用接口，对开发人员提供操纵、维护数据库数据的灵活、安全、可靠中间件支持。数据持久化层在系统中可能有 2 种不同类型的对象调用：(1)J2EE 容器中业务逻辑层的 EJB 对象；(2)UI 层的 JSP 和 Servlet 对象。

2.2 数据持久化层的总体结构

数据持久化层共由 6 个子模块组成，即数据存取模块、数据源适配合模块、异常处理模块、数据调用解析模块、数据类模块、持久化数据调用模块，在这 6 个模块中，数据调用请求由持久化数据调用模块传递到数据存取模块，数据存取模块通过调用数据调用解析模块中的调用语言解析器生成各调用语句，再根据不同的调用语句适配数据源模块，得到确定的数据源信息，执行各调用语句得到需要的数据信息；返回的数据信息封装为数据类模块中不同的 JavaBean，以 JavaBean 数组形式返回持久化数据调用模块，或者将数据调用结果的信息返回。总体结构如图 2 所示。

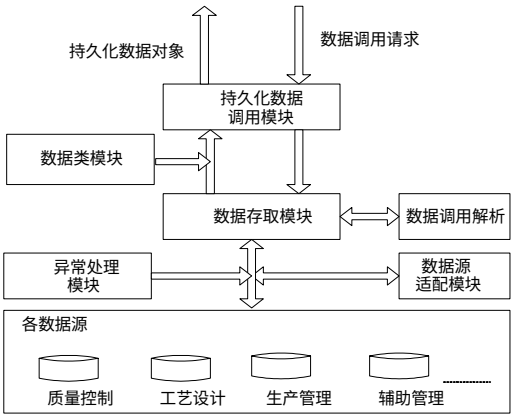

图 2 数据持久化层的总体结构

数据存取模块是数据持久化层的核心，是对不同数据源数据调用逻辑的具体实现部分；数据调用解析模块和数据源适配模块是数据存取模块准确运行的基础，通过这 2 个模块获得确定的目标数据源及其数据调用语句；持久化数据调用模块是持久化数据层与应用逻辑层连接的纽带，信息只有透过它才能进行传递，在保证数据安全的同时，减轻业务逻辑数据调用复杂性；数据类模块，其产生是数据持久化层数据调用的结果，该模块也是数据存取操作的对象；异常处理模块是系统良好运行不可或缺的部分，它把数据在存取过程中的异常转换成用户和开发人员可识别模式，并对异常进行有效的处理。

3 数据持久化层的实现

数据持久化层的各数据源在 SessionEJB 的配置文件中进行映射，通过 J2EE 连接器获得连接，数据源调用以程序代码形式实现；同时为减少平台运行时的系统资源占用率，在保证系统运行效率基础上，采取数据调用期间实例化的策略，实现平台的动态配置及管理。

3.1 数据类模块

数据类模块是一个标准的 JavaBean，在这个 Bean 中只有持久化数据对象的属性及其 set/get 方法，应用逻辑层通过持久化数据调用模块进行查询后得到的结果以 Bean 数组形式返回；同样，应用逻辑层数据更新操作也以 Bean 数组作为参数传递到数据持久化层；应用逻辑层通过 Bean 中的 set/get 方法得到或设置具体的数据，Bean 程序如下：

```
public class Qm_Bean {
  int id;
  String name;
  public int getId() {
        return id;}
  public void setId(int id) {
        this.id = id;}
  public String getName() {
        return name;
```

```
    }
    public void setName(String name) {
        this.name = name; }}
```

3.2 数据存取模块

数据存取模块是实现持久化数据调用的具体操作，并自执行数据调用语言的解析和数据源定位。数据调用方法主要分为 3 类，即查询、更新、创建，在这 3 种方法中，持久化数据的查询操作的参数是 1 个有 2 个字符串属性的 JavaBean, 一个属性是查询操作的数据名称，一个属性是此名称所代表数据特征描述；更新操作和创建操作的参数是持久化数据对象的数据。

一个数据存取模块实现如下所示：

```
public class QmDOC
{
  String[] SQLstr = null;
  String[] JNDIstr = null;
  public QmBean[] QmBeanSelect(String[] select){
…}
  public int QmBeanUpdate(QmBean[] qmArray){
…}
  public int QmBeanCreate(QmBean[] qmArray){
...}
}
```

3.3 数据调用解析模块

数据调用解析模块是特定数据源的数据存取语句生成模块，根据不同的数据调用动作，生成相应数据源的操作语句，数据调用动作有 3 种，相应的数据调用动作生成语句也是 3 种，即创建动作语句生成方法、更新动作语句生成方法、查找动作生成方法。创建、更新传递的参数是一个 Bean 数组，其动作生成语句较易于实现；难于实现的是查询动作语句生成问题，在上面所述数据调用模块中将此查询操作的调用参数定义为 1 个拥有 2 个字符串属性的 JavaBean 数组，名称属性确定相应数据源，特征属性是相应数据源数据的查询条件。

一个数据调用解析模块的实现如下所示：

```
public class ProduceSQL {
  public static String[] produceUpdate(QmBean[] qmArray){
…}
  public static String[] produceSelect(SelectStr[] select){
…}
  public static String[] produceCreate(QmBean[] qmArray){
…}}
```

3.4 数据源适配模块

数据源适配模块因为只负责适配具体数据源的名称，不负责连接操作，在数据调用过程中的创建与更新及查找动作所使用的数据源相同，所以此模块只有一个数据源适配动作。当然，为保证在拥有多个持久化数据对象时复用此方法，需要将持久化数据对象名称作为参数进行传递。

3.5 持久化数据调用模块

持久化数据调用模块是一个 SessionBean，它是业务逻辑层数据调用的规范化接口。通过在 SessionBean 中将不同的数据存取模块实例化，再套接至相应标准接口即可实现持久化数据的规范调用。

一个数据存取模块在 SessionBean 中实例化后持久化数据对象的实现如下所示：

```
public class QmSessionBean    implements SessionBean {
  public int QmCreate(QmBean[] qmArray){
  QmDOC qmdoc = new QmDOC();
  return qmdoc.QmBeanCreate(qmArray);}
  public QmBean[] QmSelect(SelectStr[] select){
  QmDOC qmdoc = new QmDOC();
  return qmdoc.QmBeanSelect(select);}
  public int QmBeanUpdate(QmBean[] qmArray){
  QmDOC qmdoc = new QmDOC();
  return qmdoc.QmBeanUpdate(qmArray);}    }
```

3.6 异常处理模块

异常处理模块是数据持久化层不可或缺的一部分，它主要通过扩展 SQLException 异常类实现特定数据库产品供应商特有异常，并以开发人员和用户能够理解的形式表现出来，以利于开发人员进行代码调试或使用户在系统运行期间准确找出出错原因，增进与系统间的沟通。不同数据源异常类的扩展方法基本相同，也没有什么技术难度，在此不再用具体实现代码予以说明。

4 结束语

本文所述数据持久化技术充分利用了使用 J2EE 开发企业应用系统过程的开发速度快、与平台无关及易于部署、移植等优点，通过使用会话 Bean 和适量的配置信息实现数据持久化，达到数据调用与应用逻辑分离的目的，在减轻部署人员工作难度及系统维护工作复杂性的同时，进一步增强了系统可移植、可扩展、可重组性能。同时，通过规范化、标准化业务逻辑层数据调用，使数据垂直传递，保证了数据调用的线程安全。目前，数据持久化技术直接应用于“棉纺织综合信息管理系统”项目的数据持久化中，整个系统处于测试、试用阶段，在 1GB 内存、2x900MCPU 服务器和 100 兆位的以太网环境下，系统可实现常用操作(与图形无关的操作)5s 内完成，峰值业务处理达 2 400 个线程，并支持并发访问用户 550 个以上。

参考文献

1 董洪彬, 窦严平. 利用 Hibernate 的 J2EE 数据持久层的解决方案[J]. 计算机工程, 2004, 30(增刊).

2 朱幸辉, 杨树强, 杜　凯. 基于构件的海量信息存储中间件的研究与实现[J]. 计算机应用研究, 2005, 22(7).

3 陈　峰, 薛士权. 通用关系数据库访问层的设计与实现[J]. 计算机工程与应用, 1999, 35 (4).

4 郑　华, 莫　林. 一种异构数据库系统集成中间件的设计与实现[J]. 微型机与应用, 2005, 24(6).

（上接第 271 页）

参考文献

1 中国音视频编码标准化工作组. AVS 大事记[Z]. 2005-12. http://www.avs.org.cn.

2 Beech D, Maloney M, Mendelsohn N, et al. XML Schema Part 1: Structures[Z]. W3C Recommendation, 2001-05.

3 Boyer J. Canonical XML[Z]. W3C Recommendation, 2001-03.

4 Gamma E, Helm R, Johnson R, et al. Design Patterns: Elements of Reusable Object-oriented Software[M]. Addison-Wesly, 1995.

5 Cargill D, Hansen J. Xerces-C++ Documentatation[Z]. 2005-05. http://xml.apache.org.