SPINE:一个轻型永久对象管理器

宋 杰 卢显良 韩 宏
(电子科技大学计算机学院 成都610015)

摘 要 在信息系统中,应用程序是基于对象模型的,但企业数据主要还是存储在关系型数据库中。永久对象管理层搭起了一个沟通对象模型和关系模型的桥梁,简化了应用程序的开发。SPINE是我们开发的一个轻型的永久对象管理构架,它允许系统中同时存在一个对象的多个拷贝,并通过版本机制来解决访问冲突;其次,它引入了三层对象设计模式,使系统更加紧凑;它还通过PROXY设计模式实现了关联对象的延迟装载,从而提高了系统性能。

关键词 永久对象,对象模型,关系模型,关系数据库,对象-关系映射

SPINE:A Lightweight Persistent Object Manager

SONG Jie LU Xian-Liang HAN Hong
(Department of Computer Science,UESTC,Chengdu 610054)

Abstract Applications used in e-busiiness of enterprise information systems are based on the object model,but the data is usually managed by relational databases. A persistent object access layer makes up the bridge between the applications based on the object model and databases based on the relational model. SPINE,a lightweight persistent object manager,allows applications to create multiple copies of a object at a time and resolve the access conflicats with the version mechanism. It introduces the 3-layer pattern to make the design of the system more compact. Moreover, the use of the prox to delay the loading of the associated objects improves the system performance dramatically.

Keywords Persistent object,Object model,Relational model,Object-relation mapping

1 引言

目前,电子商务和企业信息系统应用程序的设计开发主要采用的是对象模型,但是出于性能、保护过往系统投资和兼容性等方面的考虑,企业的数据绝大部分还是存放在关系型数据库中。然而,关系模型缺少对象模型的继承、关联以及多态等概念,这是造成系统开发复杂性的原因之一。永久对象管理层的作用就是搭起一座沟通这两个模型的桥梁,它既负责把数据库的数据转化为应用程序对象模型中的对象,也负责把这些对象转化成关系模型中的记录并存储到数据库中。永久对象管理层的出现简化了应用程序的开发,提高了它们的可移植性。

JDO与Hibernate是两个目前应用比较广泛的通用永久对象管理层。它们具有事务处理、对象缓存和对象查询语言等功能。但为了追求系统的通用性和灵活性,它们的结构相当复杂,也难于针对专门应用进行优化。同时,它们使用了JAVA语言的反射机制(Reflection),这影响了系统的可移植性。

Virtual Helpdesk是我们开发的一个P2P的远程技术支持系统。它需要处理的对象种类不多,对象-对象之间的关系也比较简单,但有其特殊的事务特性,同时系统吞吐量要求比较高。因此我们开发了专用的永久对象管理层SPINE。首先,它允许一个永久对象同时存在多个内存拷贝,并通过版本机制来解决访问冲突;其次,它扩展了文[2]的双层对象设计模式,突出了我们自己的三层对象设计模式,使系统结构更加紧凑;最后,为了提高系统吞吐量,SPINE引入PROXY设计模式,实现了关联对象的延迟装载。用户也可以通过实现自己的PROXY来实现相关对象的预装入。

虽然SPINE是一个专用的永久对象管理器,但我们相信它采用的算法和设计模式可以为类似系统的开发提供有益的借鉴。

2 对象-关系映射规则

对象-关系映射规则的主要功能是把应用程序使用的对象转化成数据库中的表。在介绍规则之前,本论文先定义几个概念。

类(class):类定义了同类对象的共同属性和基本操作。

对象(object):一个对象是一个类的具体实例。例如,对象张三是类人的一个具体实例。

关系(relationship):关系指对象和对象之间的关联。在我们的系统中,对象之间主要有两种关系:拥有(has-a)和继承(is-a)。比如对象张三拥有一个驾驶执照对象(张三 has-a 驾驶执照);轿车继承了汽车的共有属性,但同时又具有自己新的特性(轿车 is-a 汽车),此时,我们也称轿车为汽车的子类。

下面我们介绍映射规则:

规则1 如果类C_i是类C的子类,我们创建新的类C_i',它包含C和C_i的所有属性。对一棵通过继承关系构成的树,我们将重复运用此规则,直到为每个叶子节点上的类生成一个包含它所有父类属性的对应类为止。

规则2 对每一个通过规则1生成的类,我们在数据库中创建一个关系R_1。类的一个属性对应于关系的一列。此外,系统自动在关系中加入一个OID列作为主键用于唯一标识一个对象。

宋 杰 博士研究生,主要研究方向:计算机网络、网络存储、分布式操作系统等。卢显良 教授,博士生导师,主要研究方向:计算机网络、操作系统。韩 宏 博士,主要研究方向:计算机网络、软件工程。

规则3 如果 C_1 和 C_2 之间存在 $1:n$ 的 has-a 关系，在 R_2 中包含 R_1 的 OID 作为外键。

规则4 如果 C_1 和 C_2 之间存在 $n:m$ 的 has-a 关系，除关系 R_1、R_2 之外，创建关系 R_{12}，并且使用 R_1 和 R_2 的 OID 作为它的外键。

3 SPINE 系统结构

SPINE（图1）由 ObjectBroker、ConnectManager 和用户自定义的永久对象类组成。ObjectBroker 是 SPINE 的核心，应用程序通过它获取永久对象的拷贝，也通过它来永久化发生了状态变化的永久对象。除此之外，它还提供了对象缓存、查询和无用对象回收等功能。ConnectManager 管理和数据库之间的连接，同时也负责把对象层次的事务映射成数据库的事务。

下面我们详细讨论 SPINE 中永久对象的结构、同步机制和关联对象延迟装载的算法。

3.1 三层对象模式

一个永久对象需要与应用程序、对象管理器和数据库打交道。一些设计模式（比如 Fasade 模式、Envelope-Letter 模式和 Observer 模式）把上述功能分散到不同的类。这种设计比较适合不同层次的功能耦合比较松散的情况。但在 SPINE 中，这三层功能耦合得比较紧，且具有相同的生命周期。如果我们仍然把它们分散到不同的类，就会增加对象管理层的复杂度。因此我们扩展文[2]中的多层类，提出了永久对象的三层类设计模式（见图2）。

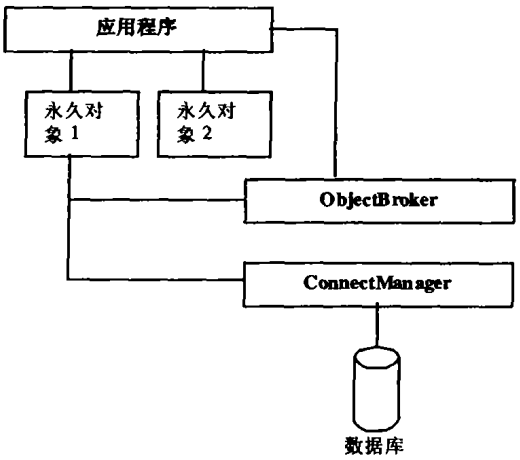

图1 SPINE 系统结构

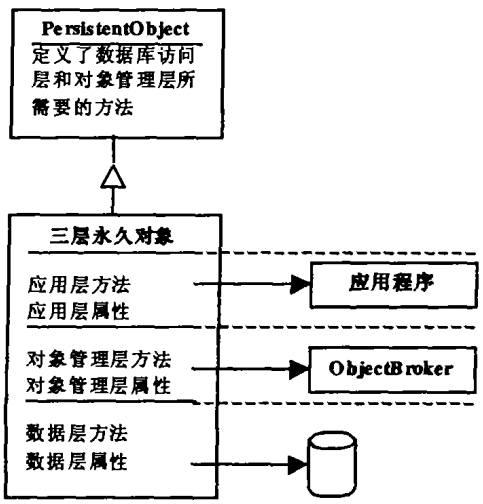

图2 永久对象的三层设计模式

在 SPINE 中，一个永久对象的所有功能被封装在一个类，这个类必须继承虚基类 PersistentObject。PersistentObject 定义了对象管理层和数据访问层所要求的方法。永久类除了实现这些方法外，还负责定义和实现与应用层有关的方法。属于不同层的方法严格地遵循强层次模型语义，也就是说，一层只允许调用紧邻它的下层所实现的方法，而不允许调用它的上层实现的方法。

3.2 多内存拷贝与版本机制

很多对象管理系统只允许一个对象在内存中有一个拷贝，所有该对象的引用都指向这唯一的拷贝，线程间的访问冲突可以通过对这个拷贝上锁来实现。这对事务一致性要求高并且访问冲突频繁的应用是适合的。

但 Virtual Helpdesk 的控制服务器有自己的特点。它访问冲突的几率很低，即使访问同一对象的线程也往往是读写对象不同的属性集合，相互间并不存在真正的数据依赖。另外，服务器经常作集合读。这样一来，单拷贝方案就不太适合了。首先，线程在访问对象时必须同步，在访问冲突几率很小的情况下，这是一种浪费。其次，即使访问同一对象的线程间不存在数据依赖，写线程仍必须锁住该对象，这影响了系统的并发度。再次，当一个线程做集合读时，集合中的对象有的处于自由状态，有的加了读锁或写锁，对象管理层必须用复杂的锁机制来支持这种操作。最后，对象管理层为了知道什么时候能释放某对象的唯一拷贝，它需要为每个拷贝维护一个引用计数。当线程不再使用该拷贝时，它必须显式地对引用计数减一。这抵消了 JAVA 自动垃圾收集的优点，容易造成资源泄露。

因此，SPINE 引入了数据库的 Multiversion 算法，实现了一个允许多拷贝的系统。在关系 R 中，除了用作主键的 OID 和对应对象属性的列，还有一个版本号列，每次修改记录它都会增加一。当线程请求一个对象时，SPINE 返回一个该对象新拷贝。各个线程可以独立地读写属于自己的拷贝。当线程希望永久化它拥有的拷贝时，SPINE 会比较拷贝与数据库记录的版本号。如果两者不同，说明已经有其它线程修改了记录，SPINE 将把这种情况通知线程。线程接到通知后，会重新请求对象的拷贝，修改新拷贝的相应属性，然后再次通知 SPINE 执行永久化操作。这种设计较好地解决了单拷贝的缺点。首先，每个线程独立操作属于自己的拷贝，避免了同步操作。再次，集合读的实现变得简单。SPINE 只需为集合中的对象创建独立的拷贝，而不用考虑该对象其它拷贝的状态。最后，系统通过 JAVA 的自动垃圾收集来回收对象拷贝，而不再需要线程显式减少引用计数。

3.3 关联对象的延迟装载

对象并不是孤立的，往往存在一系列对象与之关联。比如，汽车对象包含一个引擎对象和四个轮胎对象，雇员对象和一个公司对象相关联等等。如果我们在装载一个对象的时候把与它相关联的对象也装载进来，但应用程序又不是每次都会访问这些关联对象，这就造成了对系统资源的浪费，也延长了对象的装载时间。因此，SPINE 引入了 PROXY 设计模式。

当一个对象被装入内存时，SPINE 并不立即装入它的关联对象。相反，SPINE 在对象中植入一个 PROXY 对象。当应用程序第一次访问某个关联对象时，这个 PROXY 透明地装入关联对象。以后，当应用程序需要再次访问该关联对象时，就不再需要 PROXY 的介入了。当然，这一切对应用程序来说都是透明的。

SPINE 既允许使用系统自身提供的标准 PROXY，也允许用户定义自己的 PROXY。用户可以通过自定义的 PROXY 实现对象预装载等功能，这可以显著地提高系统的性能。

结论 SPINE 是一个为 Virtual Helpdesk 系统而开发的专用永久对象管理器。它允许同时存在一个对象的多个拷贝，并通过乐观锁来解决对对象的访问冲突；它使用三层对象设计模式把对象应用层、对象管理层和数据访问层的代码都封装在一个类之中；它还通过代理设计模式实现了关联对象的延迟装载。这些算法和设计模式可以为类似系统的设计提供借鉴。

参 考 文 献

1 Tesch T, Volz M. A Lightweight Object Manager for Group-Aware Applications: [GMD Report 47]. 1999
2 Coldwey J, Keller W. Multilayer Class. 2001
3 Keller W. Persistence Options for Object-Oriented Programs. 2004
4 Keller W. Mapping Objects to Tables - a Pattern Language. In: Proc. EuroPLoP 1997
5 Keller W. Object/Relational Access Layers - a Roadmap, Missing Links and More Patterns. 1998
6 Ambler S W. The Design of a Robust Persistence Layer For Relational Databases. White paper, 2000

（上接第76页）

和较低的误报率（平均18%）。

表1 验证算法2.1的有效性

病毒全名	W97M.NSI.C	O97M.Tristate.C	W97M.Chaos.A	VBS.Haptime.A@mm
病毒类型	宏病毒	宏病毒	宏病毒	邮件蠕虫
检测器集大小	50	50	50	10
刺激阈值	22	22	21	16
进化最大代数	1500	1500	1300	1500
检测率(10次)	80%	40%	90%	70%
误报率(10次)	20%	20%	10%	20%

注：病毒全名依据 Norton AntiVirus Virus Definitions[9]。

(2)实验2：验证算法2.2的有效性

为了检验算法2.2的改进效果，我们分别用算法2.1和算法2.2对选定的几种有代表性的宏病毒进行了多次检测实验。

实验相关参数设置如下：初始自我集大小都是8807；每代生成成熟检测器的数量为500；进化的最大代数为500代。

表2 验证算法2.2的有效性

病毒全名	算法2.1	算法2.2	自我集扩充后的大小(平均)
W97M.Aliv	40%	80%	19214
W97M.OutlookWorm.Gen	60%	90%	16443
W97M.Class.A.Gen	50%	70%	23248
W97M.Kolop	0%	40%	23924

从表2可以看出，算法2.2的检测率相对于算法2.1有一定程度的提高，这也验证了算法2.2中所作的改进是有效的。

(3)实验3：算法2.1和算法2.2的检测速度比较

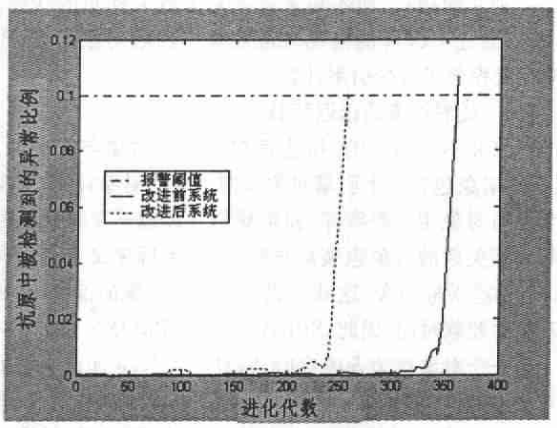

图5 W97M.Nort.A.Gen 病毒在改进前后系统中的检测曲线

本实验将考察了算法2.1和算法2.2的对这些病毒的检测速度。图5给出了算法2.1和算法2.2对其中一个宏病毒，即W97M.Nort.A.Gen 的检测曲线。

图5中改进前系统检测曲线是算法2.1的检测曲线，而改进后系统检测曲线是算法2.2的检测曲线。从图中可以看出，对于病毒 W97M.Nort.A.Gen，算法2.1需要361代的进化才能检测到，而算法2.2只需要257代。这说明了算法2.2中所做的改进使得病毒检测速度有较大的提高。

结束语 本文基于生物免疫中进化学习和阳性/阴性选择机制，提出了一种用于未知病毒检测的检测器和自我均自适应变化的免疫识别模型和算法，用以克服非我空间庞大和自我无法完整获取对检测率和误报率所产生的负面影响。通过对测试集中的几种实际病毒的检测实验，证明了该模型和算法是合理且有效的。

从本文的实践也可以看出，生物免疫中仍然有许多可以借鉴的机制等待我们去发掘，特别是对于人工免疫与网络安全研究中所遇到的问题，回到生物免疫中去寻找答案往往是直接而有效的。当然，基于人工免疫的反病毒研究还处于起步阶段，还有很多工作值得进一步研究。本文的方法也有一些应深入展开的内容，如基于大样本集的实验测试等，需要在以后的工作中逐步深入。

参 考 文 献

1 Chess D. The Future of Viruses on the Internet. http://www.research.ibm.com/antivirus/SciPapers/Chess/Future.html
2 Forrest S, Hofmeyr S, Somayaji A. Computer Immunology. Communications of the ACM, 1997, 40(10): 88～96
3 Forrest S, Perelson A S, Allen L, Cherukuri R. Self-nonself Discrimination in a Computer. In: Proc. of the 1994 IEEE Symposium on Research in Security and Privacy, Los Alamitos, CA. IEEE Computer Society Press, 1994
4 Kim J, Bentley P. An Artificial Immune Model for Network Intrusion Detection. In: Proc. of 7th European Congress on Intelligent Techniques and Soft Computing (EUFIT'99), Aachen, Germany, 1999
5 龙振洲. 医学免疫学. 人民卫生出版社(第二版), 1996
6 Burnet F M. The Clonal Selection Theory of Acquired Immunity. Vanderbilt University Press, Nashville TN, 1959
7 漆安慎, 杜婵英. 免疫的非线性模型. 上海科技教育出版社, 1998. 74～76
8 Luo Wenjian, Zhang Sihai, Liang Wen, Cao Xianbin, Wang Xufa. NIDS Research Based on Artificial Immunology. Journal of University of Science and Technology of China, 32(5): 530～541
9 http://www.symantec.com