

基于 J2EE 的 B/S 模式应用中数据访问优化设计

Optimized Design of Data Access in B/S Mode Application Based on J2EE

陈晓苏 陈小霞 吴永英 刘剑

(华中科技大学计算机科学与技术学院 湖北武汉 430074)

摘要:通过分析基于 J2EE 平台的 B/S 模式应用中数据访问效率较低的若干因素,给出了一种提高数据访问层效率的优化设计方法,并运用在基于 B/S 模式的网上阅卷系统中,较大地提高了数据访问效率。

关键词:J2EE EJB 数据访问 效率

1 引言

随着基于 J2EE 的 B/S 模式在企业级分布式应用中的使用日趋广泛,大规模分布式数据库的访问效率日益显得重要。因此,对 B/S 模式数据访问模块的优化具有十分积极的意义。

2 B/S 模式下数据访问层性能分析

基于 J2EE 的 B/S 模式如图 1 所示。其中,Web Server 层根据客户端的需求向数据访问层提交数据请求;数据访问层完成数据库的访问工作,并向其它层屏蔽数据库的访问。

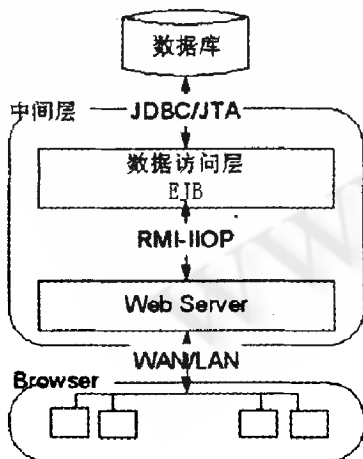


图 1 三层 B/S 架构模型图

图 1 的数据访问层中与性能相关的主要因素包括:

(1) EJB 的接口设计。EJB 提供两种访问接口:远

程接口和本地接口。目前多数分布式系统的中间层均采用多台机器提供服务,EJB 调用者只能使用远程接口访问 EJB,此时接口之间的参数以值参的形式进行传递,需进行序列化和解序列化操作,会耗费大量时间,从而影响执行效率。

(2) 数据的一致性。在与数据库的交互过程中,为同步数据库和内存中的实体对象,保证数据的一致性,必然导致大量的数据库读、写操作,从而影响整个系统的效率。

(3) 并发处理。在 J2EE 架构下,EJB 规范中严格限制了在 EJB 容器中创建新的用户线程,而且只能同步调用会话 Bean 上的方法。这些都使得查找等应用无法直接利用并发处理,因此客户端的响应时间得不到保证。

(4) 数据库连接。J2EE 支持两种数据库连接方式:直接连接和池连接。直接连接由客户端打开并维护,每次使用之后必须关闭。池连接则是由 J2EE 服务器打开和维护。在池连接模式下,J2EE 服务器启动时会建立拥有一定数量连接的连接池,调用者需要连接时,从池中申请一个闲置的连接,使用完毕,再归还给连接池,供其它调用者使用。

对于一个与数据库交互频繁的系统来说,数据访问层与数据库的连接方式如果选择不当,也会成为制约系统性能的瓶颈。

3 数据访问层优化设计的基本思想

3.1 数据访问层的基本结构

通过前面的分析,从优化数据访问层的角度出发,图

2 给出了基于若干优化策略的数据访问层基本结构。

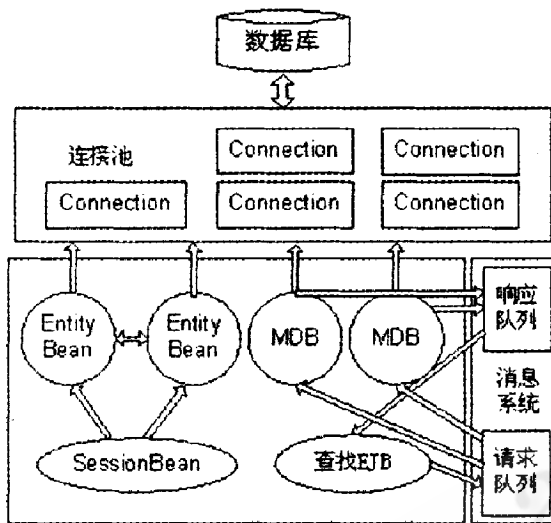


图 2 数据访问层的基本结构

在图 2 中, Web Server 通过 Session Bean 调用 Entity Bean 的本地接口, 而不直接调用 Entity Bean 的远程接口, 并由 Session Bean 封装对 Entity Bean 的访问。系统中涉及的数据对象均由对应的 Entity Bean 来封装对其的操作, Entity Bean 通过连接池与数据库服务器交互。

对于响应时间要求比较高的应用(例如实时查询), 采用 MDB 实现其并发性。对每个查询请求, Web Sever 通过查找 EJB 发出请求至请求队列, MDB 取出并处理请求, 然后将处理结果发送至响应队列, 查找 EJB 取出响应返回给 Web Server。

3.2 优化策略描述

(1) 避免采用 EJB 的远程接口。在 Web Server 与 Entity Bean 之间引入 Session Bean, 将其与 Entity Bean 部署在同一 EJB 容器中, Web Server 层不再调用 Entity Bean 的远程接口, 而通过 Session Bean 调用 Entity Bean 的本地接口, 以形参的方式传递参数, 不需对参数进行序列化和解序列化, 可大大的提高数据访问效率。

(2) Entity Bean 的持久性。通常 Entity Bean 在每次业务逻辑调用的前后调用 `ejbLoad()` 和 `ejbStore()` 方法来保持数据的持久性。实际上, 当数据没有变化时, `ejbStore()` 的写操作会导致不必要的开销, 降低系统性能。可以考虑修改 Entity Bean 的处理手段, 为其成员变

量设置一个布尔型标志位, 缺省置为 `false`, 如果成员变量的值发生了变化, 则置为 `true`。`ejbStore()` 可通过判断该标志位来决定是否进行数据库写操作。每次 `ejbStore()` 调用后都将标志位置重为 `false`。这样只有实体对象的持久性状态发生改变时, `ejbStore()` 才会对数据库进行更新操作。

(3) 利用 MDB 实现并发处理。系统中对响应时间要求比较高的部分考虑采用 MDB 实现。MDB 实现了一组新的接口, 这组接口使得 MDB 能够异步地接收和处理 JMS 消息生产者发送到队列的消息。EJB 产生消息, MDB 接收消息, 经过处理后发送回响应。一个 MDB 能够处理来自多个发送者的消息, 同时, 容器能够建立 MDB 的缓冲池, 实现消息的并发处理。

(4) 使用池连接。使用池连接时, 数据访问层通过连接池与数据库服务器交互, 这样在访问数据库时可使用 J2EE 服务器中已经建立的连接, 而无需每次都建立一个连接, 从而减少因建立连接而耗费的时间。同时, J2EE 服务器会及时收回不再使用的连接, 动态的维护连接的使用和连接池的容量, 保证了连接的利用率。

4 实现实例

网上阅卷系统利用计算机和网络支持阅卷教师完成主观试题的阅卷工作, 以提高阅卷工作的质量和效率, 保证公平性。由于该应用系统的客户端存取的数据绝大多数都是图片, 数据量和数据访问压力均很大。针对此应用特征, 本文基于前述的优化设计基本思想对网上阅卷系统的数据访问层进行了优化。

4.1 数据访问层优化实现细节

(1) 在 Session Bean 中采用 EJB 的本地接口。以网上阅卷系统中的分数实体 Score 为例, 涉及的实体对象有用户信息实体 UserInfo 和试题信息实体 PaperInfo, 三者对应的 EJB 分别为 Score、UserInfo 和 PaperInfo, 其关系如图 3 所示。Score 中封装了与试题及分数相关的成员变量和业务逻辑。Score 中的用户信息与 UserInfo 对应, 试题信息与 PaperInfo 对应。

在 Session Bean 的部署描述文件中需定义对 Entity Bean 引用之后才可以获取要访问的 Entity Bean 接口, 部分代码如下:

```
<session >
```

……//其他描述

< ejb - local - ref >

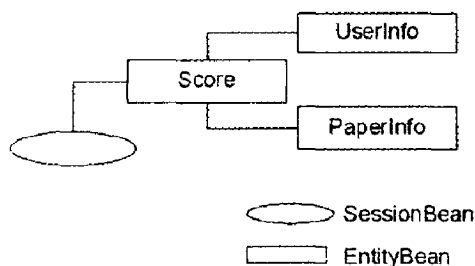


图 3 EJB 关系图

```
< ejb - ref - name > ejb/Score </ejb - ref -
name >
< ejb - ref - type > Entity </ejb - ref - type >
< local - home > ScoreHome </local - home >
< local > Score </local >
< ejb - link > Score </ejb - link >
</ejb - local - ref >
</session >
```

在 SessionBean 的实现类中,首先在 ejbCreate()方法和 ejbActivate()方法中获取 EntityBean 的 home 对象,通过 home 对象得到 EntityBean 的本地接口,在 SessionBean 的其他业务逻辑中就可以通过本地接口调用 EntityBean 的业务逻辑。部分代码如下:

```
Context context = getInitialContext();
//通过 EntityBean 的 jndi (Java 命名和目录接口)
名字找到 Home 对象
Object ref = context. lookup(" ScoreLocal" );
//获取 Home 对象
myLocalHome = ( ScoreHome) ref;
//获得 Local 接口
myLocal = myLocalHome. create ( new Integer
(1));
```

(2) 数据库的更新。对数据对象的具体操作,如获取试题图片和存分等,由各数据对象所对应的 EntityBean 的业务逻辑实现。考虑到阅卷老师获取试题图片并评阅之后,只有分数发生变化,而图片没有改变,此时 PaperInfo 中的 ejbStore()方法就不必写数据库,从而使性能得到优化。

(3) 实时查询中的 MDB 并发。实时查询阅卷质

量分析信息的处理流程为:查找 EJB 提交查找请求后,系统将请求信息封装到一个 JMS 消息中发送到请求队列, MDB 从请求队列中取出消息进行相应处理,并将结果放到响应队列,然后由查找 EJB 从响应队列中取出对应的查找结果。所涉及的实体对象有质量信息 QualityInfo、分数 Score、试题 PaperInfo、用户信息 UserInfo。对应的 EJB 关系如图 4 所示。

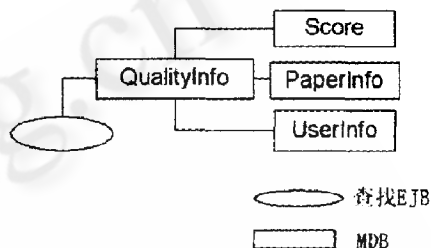


图 4 查找 EJB 关系图

图 4 中,查找 EJB 与 QualityInfo 直接交互,而 QualityInfo 封装了统计分数、试题、用户等相关信息的操作,这涉及到的 EJB 有 Score、PaperInfo、UserInfo。

(4) 连接池的配置。首先建立到数据库的连接池,连接池中封装了连接数据库的信息:数据库的 JDBC 驱动程序、数据库名字、所用账号/密码。然后建立基于该连接池的数据源。访问数据库时,通过数据源的 jndi 名字查找找到数据源,再通过数据源获得 Connection(连接对象),进行相应的数据库操作。

4.2 测试比较

(1) 测试环境。系统配置和设置:

数据库服务器配置:CPU:P IV 2.4G,硬盘 120G,内存 1G,操作系统 Windows 2000 Advanced Server。采用 20 个试题表,每个表有 15 万余道试题,每道试题的大小约为 10K 左右。

中间层服务器配置:CPU:P III 1.8G,硬盘 100G,内存 512M,操作系统为 Windows 2000 professional, J2EE 服务器采用 BEA WebLogic Platform 8.1。

客户端:操作系统统一采用 Windows 2000 professional。

(2) 测试过程。在启动客户端以及各客户端取卷的过程中,观察服务器的 CPU 占有率以及内存的使用率,同时观察客户端的实际取题速度。

(下转第 78 页)

(3) 测试结果。优化前系统客户端数目达到 100 个左右时,取题速度明显变慢,取题等待时间平均达到 3s 以上,中间层服务器的 CPU 占有率峰值达到 60% 以上。优化后的系统在 320 个客户端并发访问时取题等待时间平均降到 0.5s,中间层服务器的 CPU 占有率峰值降到了 60% 以下。

二者统计数据如表 1 所示。

表 1 测试结果

	优化前	优化后
客户端数目	100	320
CPU 占有率均值	65%	58%
平均取题时间	3000ms/题	500ms/题
平均查询响应时间	626.75 毫秒	494.51 毫秒

测试结果表明进行优化后的实现方案性能比较未经过优化的方案有显著的提升。

参考文献

- 1 飞思科技产品研发中心,赵强、乔新亮,J2EE 应用开发(Weblogic + Jbuilder),电子工业出版社。
- 2 D Roman 著,精通 EJB(第二版),电子工业出版社。
- 3 陈智罡,J2EE 应用中以消息驱动 Bean 实现并发处理,计算机系统应用,2004 年,第 12 期。
- 4 郝浩、孙旭光、郭景峰,一种提高 BMP 实体 Bean 性能的开发方法,北京电子科技学院学报. 2004 年 6 月,第 12 卷第 2 期。
- 5 倪雨婷、蒲宇,基于校园办证系统的分布式 EJB 性能分析,四川大学学报(自然科学版). 2005 年 4 月,第 42 卷第 2 期。
- 6 Ian Gorton,Anna Liu. Evaluating the performance of EJB components. Internet Computing, IEEE Volume 7, Issue 3, May - June 2003 Page(s) :18 -23.