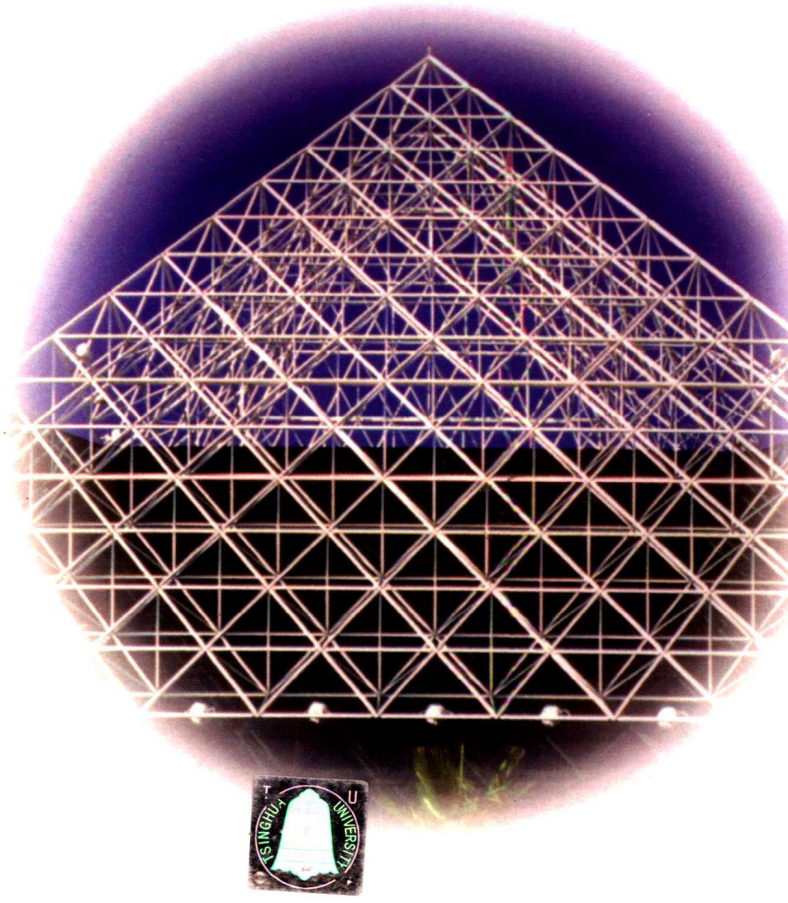
Building Highly Scalable Database Applications with .NET

构建高度可伸缩的 .NET 数据库应用程序

Wallace B. McClure
John J. Croft IV 著
李万红 译



清华大学出版社

构建高度可伸缩的.NET数据库应用程序

Wallace B. McClure

John J. Croft IV

著

李万红

译

清华大学出版社

北　京

北京市版权局著作权合同登记号　图字：01-2002-6277

内 容 简 介

本书以.NET Framework 平台为基础，详细介绍如何利用 SQL Server、Oracle、DB/2 或者 MySQL 数据库构建高度可伸缩的 C#和 Visual Basic .NET 数据库应用程序。主要涵盖了 ASP.NET 和 IIS 的可伸缩性、可伸缩性问题管理、ADO.NET 的性能因素、.NET 组件和 COM 的利用、线程、服务和 MSMQ、与各种数据库的融合、解决 RDBMS 上的典型问题和性能瓶颈等。此外，本书还带有许多实用的技巧和代码示例，可以帮助您提高解决实际问题的能力。

本书适合那些具有 C#或 Visual Basic .NET 开发经验并对 SQL Server、Oracle、DB/2 和 MySQL 有一定了解的中高级开发人员。

图书在版编目(CIP)数据

构建高度可伸缩的.NET 数据库应用程序/(美)麦克鲁尔，(美)克罗夫特著；李万红译.
—北京：清华大学出版社，2003
书名原文：Building Highly Scalable Database Applications with .NET
ISBN 7-302-06330-3
Ⅰ.构… Ⅱ.①麦…②克… ③李… Ⅲ.数据库系统—程序设计 Ⅳ. TP311.13
中国版本图书馆 CIP 数据核字(2003)第 008982 号

出 版 者：清华大学出版社(北京清华大学学研大厦，邮编 100084)
http://www.tup.com.cn
责任编辑：陈宗斌
封面设计：康博
版式设计：康博
印 刷 者：北京市清华园胶印厂
发 行 者：新华书店总店北京发行所
开　　本：787×1092 1/16　**印张**：23.5　**字数**：601 千字
版　　次：2003 年 4 月第 1 版　2003 年 4 月第 1 次印刷
书　　号：ISBN 7-302-06330-3/TP・4778
印　　数：0001～4000
定　　价：48.00 元

前　言

本书着重讨论如何创建可伸缩的应用程序。许多书都只是将重点放在特定语言或数据库产品上，而这些书在讨论时通常缺少具体场境。那些介绍编程语言的书也只是提供一些语法和基于常用算法的示例。它们都首先介绍“Hello,World”示例，然后便展开讨论。而介绍数据库的书则先介绍基本 SQL，然后再讲解专用的 DLL 和 DML 扩展，最后讲述如何使用随产品提供的管理工具。虽然这些书非常有用(这样的书堆满了我们的书架)，但是它们都只阐述单一的主题，对于更有经验的开发人员而言就不是那么有用。经验老道的开发人员学习新语言一般不需要消化一本书的全部内容，因为他们具备足够的算法知识。绝大多数开发人员都有应用 SQL 的经验，也知道应该提供什么工具。

当然可以使用技术文档，但是技术文档却不会解释如何使用.NET 创建可伸缩的应用程序。技术文档阐明与语言和环境相关的具体技术或详细信息。技术文档阐述如何做，却不说明为什么这样做。如果说这些专题类的书籍泛而不精，那么技术文档就是事无巨细地进行全面讲述。可伸缩性用于全局战略，同时也是为了选择正确的技术来实现战略。

我们都有开发和使用大型系统的经验，瓶颈会导致这些大型系统出现大量性能问题，这些都是可以预见的。不过，使用只能处理 10 条或 100 条记录的技术来处理数万或数十万条记录时，出现严重故障就会导致这些问题。因此，随着 Microsoft 的最新开发环境的推出，很有必要编写一本新型的书—— 该书阐述在解决特定问题(在本书中，要解决的问题就是创建可伸缩的数据库)时所采用的相关技术。这些资料向经验老道的编程人员提供解决问题的大量基本原理和创建大型数据库系统所需的编码，这些对他们是非常有用的。

本书提出了一些设计可伸缩系统的思路。也就是说，不需要随机应变的设计策略也可以实现可伸缩系统。通过评估要求和实现稳定的解决方案就可以满足这些需要，这样就可以实现高度可伸缩的系统。也就是说，必须知道系统主要目标，因为某些性能要求可能与其他要求不一致。单一系统不能同时全面伸缩。因此，必须设置优先级或将两个相互竞争的请求彼此隔离开。最常见的就是将 OLTP 系统与数据挖掘或 OLAP 系统隔离开。创建大型系统时，必须权衡如何在这些设计选项之间作出选择。

在这种情况下，很难做出选择，事先权衡不同设计的利弊是很重要的。一般来说，解决问题的出路不只一条，检索或更新数据也不只一种方法。独立地分析，不同的方法是否都是相同的，但是在实际系统中，技术会对其他目标产生影响。记住这一点是很重要的：我们作出的某些决定是不正确的。现在，就算人们避免提出自己的判断或只是提出意见，在给定条件下仍然会有一些数据库设计决定是不正确的。之所以这样说，是因为使用错误方法时，可以看到相应时间会增加几个数量级。

基于以上原因，我们希望阅读本书的读者都能理解可伸缩性的概念，以及可伸缩性所带来的影响。希望读者理解如何权衡不同设计并进行比较作出选择，以及为什么一些设计根本就不具备可伸缩性。尽管某些设计可以采纳，但有些设计却不应该使用。

期待该书能实现我们的初衷。

另外，必须从技术层面上选择要使用的语言和数据库，下面将给出做出这些选择的背景信息。

数据库

本书后面几章将介绍 4 种类型的数据库：

- SQL Server：选择 SQL Server 的原因很明显。它可以展示 Microsoft 的端到端解决方案，这是三大顶级企业数据库之一，.NET 代码主要就是依据它来开发的。
- Oracle：Oracle 也是一种简单选择，它是客户/服务器领域最好的数据库服务器。许多使用 Microsoft 工具的地方都会使用 Oracle 服务器。
- DB/2：尽管 DB/2 在客户/服务器领域不如 Oracle 和 SQL Server 流行，但是它作为 IBM 王牌数据库，其重要性不容低估。从微型服务器到小型机和大型机，DB/2 系统都可以容纳海量公司数据。
- MySQL：许多企业人士可能认为 MySQL 是一种开放源代码的时髦口号，仅仅用于展示开放的思想。不过，在实际的开放源代码中，MySQL 早就将自己创建成一个稳健可靠的数据存储系统。注意它在创建过程中自始至终都融入了 Web 的思想，目前可并发支持大量的 Web 站点，它的可伸缩能力已得到验证，并且其用户数量正在持续增长。

虽然还有其他优秀的数据库可以使用，但在此不能一一进行讨论。我们相信该书将会把这 4 种最重要的数据库讲清楚。其他技术(如 OODBMS 或更新的 XML 数据库)也是用于这些特定目的。如果人们要管理数百万条数据记录，就可以选用关系数据库系统。

语言

本书绝大多数代码都是采用 Visual Basic.NET 或 C#编写的。为什么要重点讲述 Visual Basic .NET 和 C#呢？Visual Basic .NET 是 VB 的进一步扩展，它是目前用于 RAD 和客户端应用程序的最流行的工具。C#提供了 C 语言式的语法以及 VB RAD 功能和非常稳健的对象库。这样，C#就具有 RAD 功能，绝大多数高级编程人员都使用这种格式。托管的 C++又如何呢？使用托管代码，就没有非托管代码所具有的编译和解释性代码。托管代码可以方便地完成许多操作。托管代码的功能局限在 C++功能之内。因此，绝大多数人都不会采用它。使用 JScript 如何呢？与此相似，JScript 没有大量解决方案可供使用，尤其是编写可伸缩的企业级应用程序时，可供使用的解决方案则更少。

本书层次结构

本书由 3 部分组成，其大纲如下：

第 I 部分：引言

第 I 部分介绍.NET Framework 的体系结构，以及如何使用该体系结构创建高度可伸缩的应

用程序。第 1 章讨论了当前的开发要求，然后阐述.NET Framework 如何帮助解决那些问题。第 2 章简要阐述.NET Framework 的概念。

第 II 部分：设计数据库和中间层组件以获得最大可伸缩能力

第 II 部分简要阐述开发使用数据库的应用程序的方法。第 3 章简要介绍数据库，以及保存数据到数据库和检索数据库中数据的策略。第 4 章概要讨论数据库如何支持事务处理以及不同事务处理和锁定级别的影响。第 5 章介绍如何使用 ADO.NET 访问一般数据库，以及如何在.NET Framework 中使用 ADO 2.X。第 6 章讨论如何编写可复用组件。第 7 章介绍如何集成 COM 组件和.NET 应用程序。第 8 章讨论在应用程序中如何正确实现线程，以及如何使用.NET Framework 正确编写 Windows 服务。第 9 章重点讨论如何通过.NET Framework 访问 MSMQ。MSMQ 是 COM+ Services 的组成部分。应用程序间可以异步传送消息。

第III部分：特定数据库伸缩问题

第III部分讲述.NET Framework 中如何访问和使用 SQL Server、Oracle、DB/2 和 MySQL 数据库。另外，还探讨了访问数据库、在数据库中插入数据、取回主键的不同方法以及其他数据库优化等问题。

附录

本书附录包含编程的一般背景信息。附录 A 概要地介绍示例应用程序。附录 B 概要地介绍编码时采用的标准。附录 C 包含.NET Framework 和 ASP.NET 最常用的在线资源列表。

目　　录

第 I 部分　引　　言

第 II 部分 设计数据库和中间层组件以获得最大可伸缩性

第III部分　特定数据库伸缩问题

第 I 部分　引　　言

第1章　当前软件开发的问题

本章主要内容：

- 理解当前软件开发中面临的挑战
- 了解 Visual Studio 6 的优缺点
- 认识.NET Framework 的一些特性如何为开发人员提供帮助

在当今的软件开发环境中，软件开发人员同时开发几种应用程序的情况是司空见惯的，因此，开发这些应用程序所用到的工具都应得到相应的支持。根据当前所开发项目的不同，可以开发出一个 C/S(客户/服务器)应用程序，Web 站点应用程序，一个中间层有 COM+服务功能的 N 层分布式应用程序，或者使用 XML 开发数据交换技术；还可能开发需要与其他平台通信的应用程序，例如：AS/400、大型机、或者基于 Unix 的系统，也可能是这些的任意组合。

Microsoft 的 Visual Studio 6 是目前最流行的优秀开发工具包。这些工具广泛用于开发很多这样的数据库。虽然孤立地看每个工具好像缺少一些关键的因素从而限制了它们应用，但这些工具相当好用，问题出在 Microsoft 的 Visual Studio 6 的开发工具包使得在一个环境中开发上述的各种应用程序变得困难起来。不是说不能用 Visual Studio 6 来开发这些应用程序，只是因为用它来开发这些类型的应用程序会更困难。

在这一章，我们将一起回顾一下前一版的 Visual Studio 6 的开发环境，同时也了解一下 Visual Studio.NET 和.NET Framework 如何能够满足开发人员的新开发需求。

1.1　Visual Basic 6

Visual Basic 6 在 Visual Studio 6 中几乎是无所不能的。利用 VB，开发人员能够轻而易举地开发 C/S(客户/服务器)和前端应用程序、中间层 COM+应用程序以及分布式组件。VB 使得开发人员能够快捷、简单地开发客户应用程序。之所以能够容易地开发这些应用程序，得益于它有一个快速应用程序开发(RAD)环境。它并不要求开发人员知道或者理解怎样在屏幕上放置一个按钮或一个文本框，甚至不必懂得如何驱动图形用户界面(GUI)以及启动初始窗体的内部机理。仅仅需要知道怎样将用户界面(UI)元素单击和拖放到窗体上。开发人员在作了这些设计之后，就可以根据事务逻辑开始编写代码，进一步完善应用程序。

正是由于这些特点，VB 被普遍认为是当前效率最高的开发语言。除了它的高效性，VB 也是一门应用广泛的语言，全世界每天大约有 350 万的 VB 编程人员使用 VB 来开发应用程序。而且，VB 编程语言还存在大量的第三方支持，您可以到互联网上去看看，会发现，大量的第

三方开发了数目众多的组件来协助开发人员开发基于 VB 的应用程序。

但是，用 VB 进行应用程序的开发也存在一些潜在的问题。问题之一是错误地应用 VB 中的一种功能，即数据绑定控件。数据绑定控件允许编程人员方便、快捷地和数据库中的记录相连接。虽然对开发人员来说，这是一个很重要的功能，但是，应用数据绑定控件会对数据库的性能产生巨大的影响。很多开发人员创建带有数据绑定控件的应用程序时，用一个字符串来创建连接。每加载一组数据绑定控件，就会创建一个连向数据库的不同连接。一个使用 5 组不同数据绑定控件的应用程序会创建 5 组不同的与数据库的连接。如果运行这个应用程的 10、50 或是 100 个实例，很容易使您的数据库服务器陷于瘫痪。这主要取决于是否允许对数据进行更新，不允许更新的潜在后果是整个数据库记录会被锁定，其他用户无法使用。开发人员如果使用数据绑定控件，很快就会陷入麻烦之中。

注意：

难道所有的数据绑定控件都是不好的吗？绝对不是。您可以用 ActiveX Data Objects(ADO)与记录集断开连接来解决这个问题。但问题是很多开发人员并不知道有非连接的记录集而没有使用它。

VB 的另一个缺点与它的功能集相关，它缺少两个其他开发语言都包含的重要特征：多线程和可视化继承。多线程功能允许将一个应用程序的工作负载分为几个部分，这意味着编程人员可以将长时间运行的任务用单线程运行从而改善最终用户的体验。通过支持在一个应用程序内执行多线程，可执行程序对用户的输入回应会更为灵敏。由于不支持多线程，用 VB 编写的应用程序其交互性不如那些用支持多线程的语言编写的应用程序。可视化继承允许编程人员为应用程序创建一个或者是几个窗体，这些标准窗体可以与应用程序的窗体相互连接，提供一个标准、一致的应用程序用户界面。通过支持可视化继承，开发语言可以使得应用程序的维护更加容易，用户界面更为一致。

这并不是意味着没有多线程和可视化继承总是件坏事。举例来说，线程是一个相对简单的概念，但是，调试多线程的应用程序非常复杂。当您开始编写这样几个线程：其中每个线程执行相同的功能，并且这些线程都必须访问开始线程中的相同变量时，您会怎样做？对这些变量的访问必须进行锁定，变量必须更新，接着再给这些变量解锁，这并不是一个复杂的概念。但不幸的是，很多 VB 编程人员对此并不熟悉。线程和继承仅仅是编程语言的特性，在这种情形下，它们是一门编程语言的复杂特性。这些特性需要编程人员花费大量的时间才能正确掌握。有了.NET Framework 和 Visual Studio.NET，绝大多数线程的问题对编程人员来说被隐藏到后台。如果需要用 Visual Basic.NET 来开发一个使用线程的应用程序，Visual Studio.NET IDE 提供了全方位的设计、开发和调试的支持功能。

用 Visual Basic 6 来开发中间层 COM+应用程序是一件极其容易的事情。只要打开 COM+ Services 功能，中间层的事务逻辑就能够被快速、容易地开发、调试和部署。用 Visual Basic 6 开发 COM+应用程序的问题在于：VB 6 运行库不支持对象池，所谓的对象池是这样一个机理：即在内存中存放目标，以便应用程序调用。由于 VB 6 的运行库对线程模型缺少适当的支持，所以也不支持对象池，而由于不支持对象池，VB 6 的 COM+组件的执行效率要比支持对象池语言的 COM+组件的执行效率要低 30%~50%。

尽管缺少对对象池的支持，VB 允许方便、快捷地开发组件，并进行调试。总的来说，这在一定程度上弥补了缺少对对象池的支持的缺点。但是如果用.NET Framework 的话，用 Visual Basic 语言开发的组件则可以访问所有的 COM+服务，包括对对象池的支持。

用 VB 6 编写的组件能方便地创建和发布。问题在于：对这些组件必须进行安装，并对现有的组件进行升级。大多数时候，这些组件的初始安装是极其容易的。但是，总有些时候，必须关闭应用程序才能对这些组件进行更新，用户并不喜欢这样，但是，他们可以容忍这些少量的用于更新的停用时间。而另一个问题：组件的更新可能会导致不兼容的问题，这才是用户真正不喜欢的。.NET Framework 提供了相对容易的应用程序安装机制，举例来说，ASP.NET 应用程序是使用.NET Framework 编写的，并没有用到.NET 环境外的一些特性，比如说 COM+等，这样，只需把这些应用程序复制到一个目录，不用安装这些组件，或者是运行任何注册工具来对组件进行注册。

1.2　Visual C++ 6

DOS、Windows、OS/2 以及其他一些环境的 C 和 C++开发工具曾一度风靡一时。Microsoft 的 Visual C++继续扮演 Microsoft 高端开发工具的角色，Visual C++ 6 是 Visual Studio 6 中的高端开发工具。相对于 Visual Studio 6 中的其他工具而言，开发人员可以运用 VC++在开发过程中的任一阶段创建更高性能的应用程序。另外，C/C++开发工具被认为是开发环境的最高阶梯。运用 VC++，开发人员能够在他们的前端应用程序中利用多线程的优点，也可以在中间层运用 COM+ Services 对对象池进行支持。有了这两个法宝，今天的 VC++开发人员可以在 Windows 2000 上开发高效的应用程序。问题是用 C/C++进行开发要比用那些快速应用程序开发工具，如：VB、Delphi 或是 PowerBuider 需要更多的开发时间。而且，由于缺少精通 C/C++的开发人员，使得很多项目并没用 C/C++来进行开发。

为什么会缺少精通 C/C++的开发人员呢？有两个原因：对指针用法(包含另一变量在内存中地址的变量)的困惑，以及在 Windows 下进行 C/C++开发的困难和专业开发人员的缺乏导致了开发成本的上涨。虽然商业应用程序开发人员能够判断用 C/C++方法进行开发的成本，包括为潜在的成千上万的客户而展开的额外开发成本，但是业务应用程序的自主开发费用是很难判断的。总的来说，这些应用程序最多仅仅分发到数千个用户手中。除了成本，开发和部署应用程序的时间也必须考虑在内。在一个业务单位寻求解决方案的时候，问题往往是无法控制的。业务单位需要方案立即解决问题。通过尽快地开发和部署解决方案，业务单位可以比竞争对手得到更多的优势。

Microsoft 为当前正在使用 Visual C++的编程人员提供了 Visual C++ .NET。对这些编程人员来说，他们喜欢 C 式语法，但是不想和复杂的指针纠缠不清，也不想学习 Windows 的开发方法。Microsoft 提供了一门新的基于 C/C++语法的编程语言，它叫做 C#(发音为“C Sharp”)。C#提供了 RAD 式的应用程序开发方式，同时也允许编程人员用他们所希望的 C 式语法。除了对 RAD 环境的 C 式编程提供支持外，C#和 Java 编程语言有很多相似之处，这使得 Java 专业编程人员可以快速、容易地转向用 C#开发。