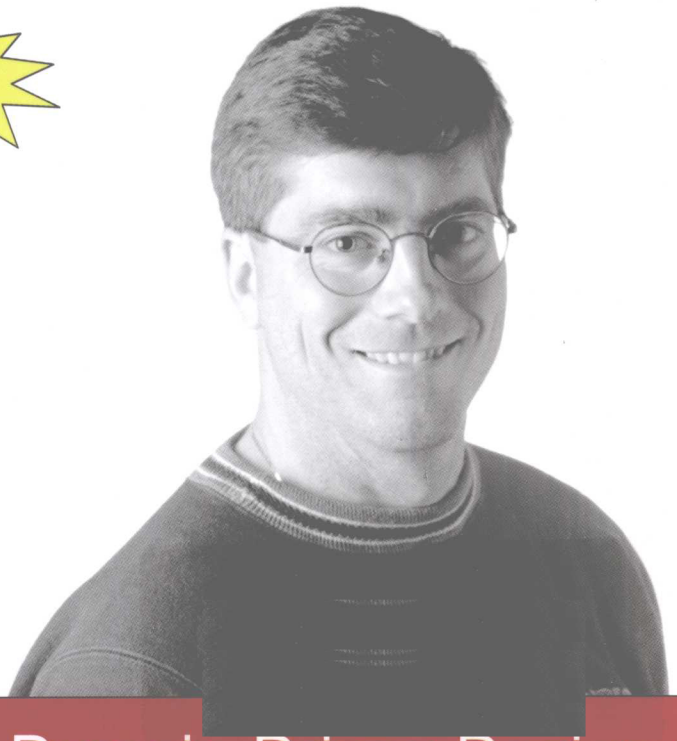
PROGRAMMER TO PROGRAMMER™

UML China
特别推荐



.NET Domain-Driven Design with C#:
Problem-Design-Solution

领域驱动设计
C# 2008实现
问题·设计·解决方案

(美) Tim McCarthy 著
UMLChina 译

wrox™
WILEY
清华大学出版社

领域驱动设计
C# 2008 实现

(美) Tim McCarthy 著
UMLChina 译

清华大学出版社
北 京

Tim McCarthy
.NET Domain-Driven Design with C#: Problem-Design-Solution
ISBN: 978-0-470-14756-6

图书在版编目(CIP)数据

领域驱动设计 C# 2008 实现/(美)麦卡锡(McCarthy，T.)著；UMLChina 译.
—北京：清华大学出版社，2010.3
书名原文：.NET Domain-Driven Design with C#: Problem-Design-Solution
ISBN 978-7-302-22191-3

Ⅰ.领… Ⅱ.①麦… ②U… Ⅲ.C 语言—程序设计 Ⅳ.TP312

中国版本图书馆 CIP 数据核字(2010)第 030098 号

责任编辑：王 军 谢晓芳
装帧设计：孔祥丰
责任校对：成凤进
责任印制：王秀菊
出版发行：清华大学出版社 **地 址**：北京清华大学学研大厦 A 座
http://www.tup.com.cn **邮 编**：100084
社 总 机：010-62770175 **邮 购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京鑫海金澳胶印有限公司
经 销：全国新华书店
开 本：185×260 **印 张**：24.25 **字 数**：590 千字
版 次：2010 年 3 月第 1 版 **印 次**：2010 年 3 月第1 次印刷
印 数：1～4000
定 价：49.00 元

本书如存在文字不清、漏印、缺页、倒页、脱页等印装质量问题，请与清华大学出版社出版部联系调换。联系电话：(010)62770177 转 3103 产品编号：030077-01

关 于 作 者

Tim McCarthy 是一名自由职业的咨询师，他采用最新的 Microsoft 平台和技术设计和构建分层的 Web 和智能客户系统。Tim 是一个解决方案架构方面的 Microsoft MVP，他精通各种各样的 Microsoft 技术，不仅局限于以下技术：.NET Framework (ASP.NET/智能客户/VSTO/工作流/Web 服务、Windows Presentation Foundation)、SQL Server、Active Directory、MS Exchange 开发、UDDI、SharePoint 和面向服务架构(SOA)应用。Tim 不仅是一个项目技术主管/成员，而且在一些财富 500 强公司担任技术咨询。他几年前已经获得 Microsoft 认证的解决方案开发人员(MCSD)和 Microsoft 认证的培训师(MCT)证书，也是首次获得 Microsoft 认证的.NET 应用开发人员(MCAD)和.NET MCSD 证书的开发人员之一。他也获得了 Microsoft 认证的 SQL Server 2000 数据库管理员证书。Tim 还是一位 IEEE 认证软件开发专家，世界上只有 550 人持有此证书。

Tim 是 Wrox 出版社的几本书的作者和审稿人。他的其他书籍包括 *Professional VB 2005*(主要作者)，几个版本的 *Professional VB.NET*，*Professional Commerce Server 2000* 和 *Professional ADO 2.5 Programming*。他还编写和拍摄了名为 *SharePoint Portal Services Programming 2003* 的 DVD。Tim 为 Developer .NET Update 新闻组撰写了大量文章、为 Microsoft Developer Network(MSDN)开发了打包的演示文稿，并且为在.NET 中使用 COM+ 服务编写了一本白皮书。他也是 *SQL Server Magazine* 和 *Windows & .NET Magazine* 的撰稿人。

Tim 在全世界的技术会议和几个圣地亚哥区域的用户组(包括.NET 和 SQL Server 组以及几个 Code Camp)上发表过演讲，过去几年，他经常在圣地亚哥的 Microsoft 开发者日会议发表演讲。Tim 也开发了不少 MSDN Webcast，其中很多是 Microsoft 多次应邀开发的。他也为企业专业的.NET 监督和培训讲授定制的.NET 课程。

Tim 拥有伊利诺斯理工大学市场营销专业工商管理学士学位以及国立大学(National University)的市场营销专业工商管理硕士学位。在成为一名应用开发人员之前，Tim 是一名美国海军陆战队的职员。Tim 对于.NET 的热情仅次于他对 Notre Dame Athletics 的热情。

Tim 的 Email 联系方式是：tmccart1@san.rr.com。

前　言

在阅读了 Eric Evans 的书 *Domain-Driven Design, Tackling Complexity in the Heart of Software* 之后，我设计软件系统的方式完全改变了。之前，我常常以一种侧重于以数据为中心设计软件对象模型，我没有真正聚焦于如何把行为和数据结合到对象中。我对这种新的思考方式感到如此震惊，因此我开始试图找到能够揭示 Eric 书中概念的代码示例。我通常用 Google 搜索我的领域驱动设计(DDD)问题的答案，并且确实发现有些东西对我有帮助，但我依然渴求更多关于这个主题的知识。

我不得不搜索.NET 中的 DDD 答案，因为 Eric 的书是与技术无关的。本书的要点是架构概念。里面到处有 Java 和 Smalltalk 的代码示例，但不是关于架构概念的。然后，就有了 Jimmy Nilsson 的书 *Applying Domain-Driven Design and Patterns*，此时我开始看到可以用于连接 DDD 概念的更多模式。Jimmy 把来自 Martin Fowler 的优秀书籍 *Patterns of Enterprise Application Architecture* 的一些概念结合起来，并说明它们如何有助于得到好的 DDD 设计原则。Jimmy 也做了大量的工作，在他的书中提供了大量优秀的.NET 代码示例，并引导读者体验了几种按照 DDD 实现操作的方法。就在我刚看完 Jimmy 的书后，我开始订阅 Yahoo! Groups 上的 DDD RSS Group feed，这也给了我很多帮助。在 DDD 小组中，我发现的一件事情是人们一直在寻找一个能够揭示 DDD 原则的.NET 相关应用。在阅读了这些帖子之后，我决定撰写这本书，把我懂得的关于如何使用 DDD 技术构建.NET 应用的知识分享给开发人员社群。我猜我可能感觉有点内疚，因为我读了小组里如此多其他人的帖子，但我只是偶尔发帖。现在，我编写了一本书代替发帖！可能这将促使我更多地融入这个小组。

我写这本书的主要目的是吸取 Eric、Martin 和 Jimmy 书中的思想和模式，构建一个真实的端到端.NET 应用。我确实希望说明我如何使用 DDD 原则在.NET 中构建一个领域模型的某些想法，但是，我并非只是构建老的.NET 应用，我也要尝试一些 Microsoft 在构建应用方面的最新技术，如 Visual Studio 2008 和.NET 3.5 框架。

本书读者对象

本书针对希望精通面向对象的设计技巧和学习 DDD 的有经验的.NET 开发人员。如果您不在这个级别，也是可以的，但我建议您至少有一些编写.NET 代码或者 Java 代码的经验。如果您之前没有编写过任何.NET 代码，则本书可能稍微有点难。

我也推荐您阅读之前提到的 Eric Evans、Jimmy Nilsson 和 Martin Fowler 的书籍。您不一定要读，但我强烈推荐阅读，因为这有助于您更好地理解本书的许多设计和模式。

因为本书的每一章都建立在前一章的基础上，所以我建议您按顺序阅读。

本书涉及范围

第 1 章，“介绍项目：SmartCA 系统”——本章介绍了我正在构建的系统——SmartCA 系统。本章不仅列出了遗留系统的问题和新系统的需求，而且说明了计划用来满足所有需求的技术和设计。

第 2 章，“设计分层架构”——本章涵盖了本书其余部分要使用的架构基础。本章介绍了一些模式，包括分层超类型模式、分离接口模式以及模型-视图-视图模型模式。我也标识和解释了几个重要的 DDD 概念。从本章我开始编写系统的代码，重点放在基础设施层。

第 3 章，“管理项目”——本章开始实现系统中管理项目的功能。本章也讨论了承包商的概念以及他们与项目之间的关系，并介绍了模型-视图-视图模型模式的第一个代码迭代。

第 4 章，“公司和联系人”——本章定义和建模了公司、联系人和项目联系人。本章也说明了如何保存不是它们自己的聚合根的实体，例如，在项目聚合内如何保存项目联系人。最后，本章介绍了在 UI 中显示和编辑值对象的技术。

第 5 章，“递交传送”——本章介绍了建筑行业中的递交传送的概念，然后使用该概念来建模递交聚合。我给领域层和基础设施层都添加了一个新概念，阐述了如何保存来自实体根仓储的子集合。本章也讨论了如何使用 Xceed DataGrid 控件来构建用户控件。

第 6 章，“信息请求”——本章介绍了建筑行业中信息请求(RFI)的概念。我也在领域中引入了一个新的模式，称为说明模式。本章也针对如何处理传送的仓储和视图模型进行了一些重要的重构。

第 7 章，“提案请求”——本章介绍了建筑行业中提案请求的概念。本章开始给领域模型添加更多的行为，并说明更丰富的领域模型类。本章也讨论了如何处理领域模型类内部违反的业务规则，并绑定说明功能。

第 8 章，“更改单”——本章介绍了建筑行业中更改单的概念。本章继续给领域模型类添加更多的行为，继续开发更丰富的领域模型类。本章介绍了两个重要的接口，IEntity 接口和 IAggregateRoot 接口。这导致了遍及整个领域模型的大量好的重构。最后，本章创建了一些更加高级的说明类。

第 9 章，“施工更改指令”——本章介绍了建筑行业中施工更改指令的概念，并进行了大量重构，主要聚焦于各个视图模型类。本章说明了接口和泛型结合的功能。

第 10 章，“和服务器同步”——本章设计和实现了如何和服务器同步客户的离线数据。本章不仅介绍了如何在客户上存储事务消息，而且介绍了如何把这些客户上的消息和服务器上的消息同步。此外，还说明了如何确保在领域模型中实现所有的同步逻辑。

第 11 章，“客户会员管理系统”——本章创建了客户会员管理系统，从而说明了如何允许用户能够在一个离线场景中执行与会员管理相关的任务。这涉及一个非常丰富的领域模型，用来表示用户及其会员管理数据，以及使用提供者而不是仓储来和数据存储交互的新概念。本章也说明了如何利用第 10 章的同步代码。

本书结构

本书本质上是一个非常大的示例研究。全部章节从头到尾构建了一个完整的系统。每一章的结构都是相同的，总体上是一个自包含的模块，由问题、设计和解决方案组成，解决方案为正在构建的系统添加了一些新的功能，最后在该章的末尾给出了一个总结。

大多数时候，问题部分相当短，而设计部分和解决方案部分占了大部分篇幅。解决方案部分总是包含实现设计的代码。

本书环境

为了运行本书的所有代码示例，您需要安装 Visual Studio 2008(安装时包括了.NET 3.5 Framework)。我强烈推荐您使用 Visual Studio 2008 Professional Edition，这样您可以运行作为代码基的一部分的所有单元测试。

另外，您需要安装以下应用程序和组件：

- SQL Server Compact 3.5(SQL CE)——免费从 Microsoft SQL Server 的 Web 站点下载(www.microsoft.com/sql/editions/compact/downloads.mspx)。
- Xceed DataGrid Control for WPF 1.3 版本——免费从 Xceed 的 Web 站点下载(http://xceed.com/Grid_WPF_New.html)。
- 某个版本的 SQL Server 2008——如果您要使用 SQL Server Management Studio 来更改 SQL CE 数据库，则这是必需的。Express Edition 可以免费从 www.microsoft.com/sql/2008/prodinfo/download.mspx 下载。

源代码

在读者学习本书中的示例时，可以手工输入所有的代码，也可以使用本书附带的源代码文件。本书使用的所有源代码都可以从本书合作站点 http://www.wrox.com/ 或 www.tupwk.com.cn/downpage 上下载。登录到站点 http://www.wrox.com/，使用 Search 工具或使用书名列表就可以找到本书。接着单击本书细目页面上的 Download Code 链接，就可以获得所有的源代码。

注释：

由于许多图书的标题都很类似，因此按 ISBN 搜索是最简单的，本书英文版的 ISBN 是 978-0-470-14756-6。

在下载了代码后，只需用自己喜欢的解压缩软件对它进行解压缩即可。另外，也可以进入 http://www.wrox.com/dynamic/books/download.aspx 上的 Wrox 代码下载主页，查看本书和其他 Wrox 图书的所有代码。

勘误表

尽管我们已经尽了各种努力来保证文章或代码中不出现错误，但是错误总是难免的，如果您在本书中找到了错误，例如拼写错误或代码错误，请告诉我们，我们将非常感激。通过勘误表，可以让其他读者避免受挫，当然，这还有助于提供更高质量的信息。

请给 wkservice@vip.163.com 发电子邮件，我们就会检查您的反馈信息，如果是正确的，我们将在本书的后续版本中采用。

要在网站上找到本书英文版的勘误表，可以登录 http://www.wrox.com，通过 Search 工具或书名列表查找本书，然后在本书的细目页面上，单击 Book Errata 链接。在这个页面上可以查看到 Wrox 编辑已提交和粘贴的所有勘误项。完整的图书列表还包括每本书的勘误表，网址是 www.wrox.com/misc-pages/booklist.shtml。

P2P.WROX.COM

要与作者和同行讨论，请加入 p2p.wrox.com 上的 P2P 论坛。这个论坛是一个基于 Web 的系统，便于您张贴与 Wrox 图书相关的消息和相关技术，与其他读者和技术用户交流心得。该论坛提供了订阅功能，当论坛上有新的消息时，它可以给您传送感兴趣的论题。Wrox 作者、编辑和其他业界专家和读者都会到这个论坛上来探讨问题。

在 http://p2p.wrox.com 上，有许多不同的论坛，它们不仅有助于阅读本书，还有助于开发自己的应用程序。要加入论坛，可以遵循下面的步骤：

(1) 进入 p2p.wrox.com，单击 Register 链接。

(2) 阅读使用协议，并单击 Agree 按扭。

(3) 填写加入该论坛所需要的信息和自己希望提供的其他信息，单击 Submit 按扭。

(4) 您会收到一封电子邮件，其中的信息描述了如何验证账户，完成加入过程。

注释：

不加入 P2P 也可以阅读论坛上的消息，但要张贴自己的消息，就必须加入该论坛。

加入论坛后，就可以张贴新消息，响应其他用户张贴的消息。可以随时在 Web 上阅读消息。如果要让该网站给自己发送特定论坛中的消息，可以单击论坛列表中该论坛名旁边的 Subscribe to this Forum 图标。

关于使用 Wrox P2P 的更多信息，可阅读 P2P FAQ，了解论坛软件的工作情况以及 P2P 和 Wrox 图书的许多常见问题。要阅读 FAQ，可以在任意 P2P 页面上单击 FAQ 链接。

致　谢

首先，我要感谢我的家人。我的妻子 Miriam 和我的女儿 Jasmine——感谢你们忍受我撰写这本书花费的大量时间。我做每一件事情时，心里总是惦记着你们!我爱你们两个!Jasmine，我想有一天你也会写书！感谢我的爸爸妈妈，您们教给我勤奋工作、永不放弃、始终自信的价值观——感谢你们，我爱你们！

我也要感谢开发编辑 Christopher Rivera，他做了大量的工作，使我免于各种杂事的干扰，潜心工作，而且他一直在支持着我，感谢 Christopher!感谢技术审校者 Doug Holland，和您一起工作很快乐，希望过一段时间还能和您合作另一个项目，感谢您！感谢采编 Katie Mohr——Katie，感谢您坚持不懈地支持我撰写这本书的想法并努力完成这本书！还有，感谢您温馨的鼓励话语，它让我时刻感到很欣慰。

我也要感谢那些在 InterKnowlogy 工作的朋友们。感谢 Kevin Kennedy——Kevin，感谢您花时间为 WPF 表单设计了一个优美的布局，您在 WPF 上花费 15 分钟，顶得过我花几个小时!非常感谢 Dale Bastow 和 Staci Lopez——感谢您对我如此有耐心，并且总是花费大量的时间来教我建筑行业的知识。感谢 Dan Hanan 和 John Bowen——感谢您们在我有 WPF 方面的问题时总是能够抽出时间，我真的非常感激。另外顺便说一下，不要忘记覆盖 GetHashcode 和 Equals。感谢 Tim Huckaby——Tim，无论我写什么感兴趣写的东西感谢您总是鼓励我，您永远是我的好朋友。

—Tim

目　录

第 1 章

介绍项目：SmartCA 系统

本书的项目基于一个真实公司的真实系统。出于隐私，公司和系统的名字作了更改。虚构的公司名是 Smart Design，系统名是 SmartCA。Smart Design 是一家正在成长的建筑、工程和室内设计企业。该公司提供很多服务，其中之一是施工管理，就施工管理来说主要是文档管理、成本控制和项目组合管理。

1.1 问题

为了管理其施工管理(Construction Administration，CA)数据和流程，10 年来，Smart Design 一直使用一个自己开发的 Microsoft Access 数据库系统，称为 Construction Administration Database，该数据库系统运行在其公司网络上。公司已经逐渐习惯了这个系统，包括它的优点和缺点。最初编写该系统时，只有少量的用户，需求非常简单，因为他们已经有 Microsoft Office 的许可证，所以开支很小。在这种情况下，使用 Microsoft Access 是一个很好的技术选择。图 1-1 显示了该系统的主屏幕。

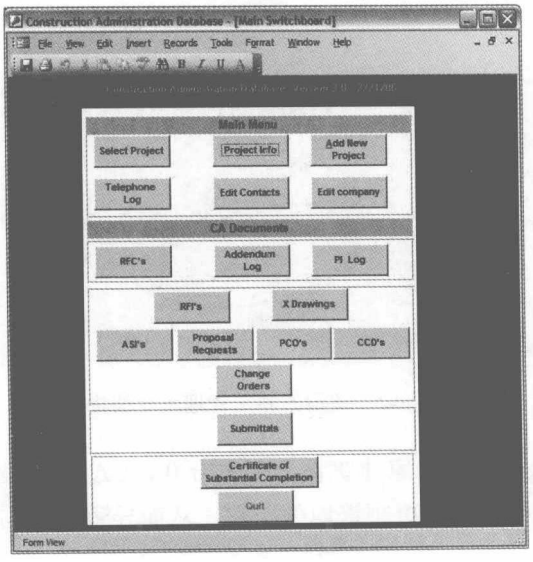


图 1-1　遗留施工管理系统的主屏幕

这些年过去了，这个系统变得越来越重要。它被修改了很多次，从代码、设置到用户界面(UI)设计。UI 表单里嵌入了许多逻辑，还有一些逻辑嵌入到数据库查询中。这个系统现在基本上成了一个智能客户(Smart Client)反模式的范例。

说明：

智能客户反模式被 Eric Evans 定义为“把所有业务逻辑放进用户界面。把系统分解成小函数，作为分离的用户界面实现，并在里面嵌入业务规则。使用关系数据库作为共享的数据仓储。使用现有的自动化程序最高的 UI 构建技术和可视化编程工具”(请参考清华大学出版社引进并出版的《领域驱动设计——软件核心复杂性应对之道》)。

图 1-2 展示了当前系统的架构。

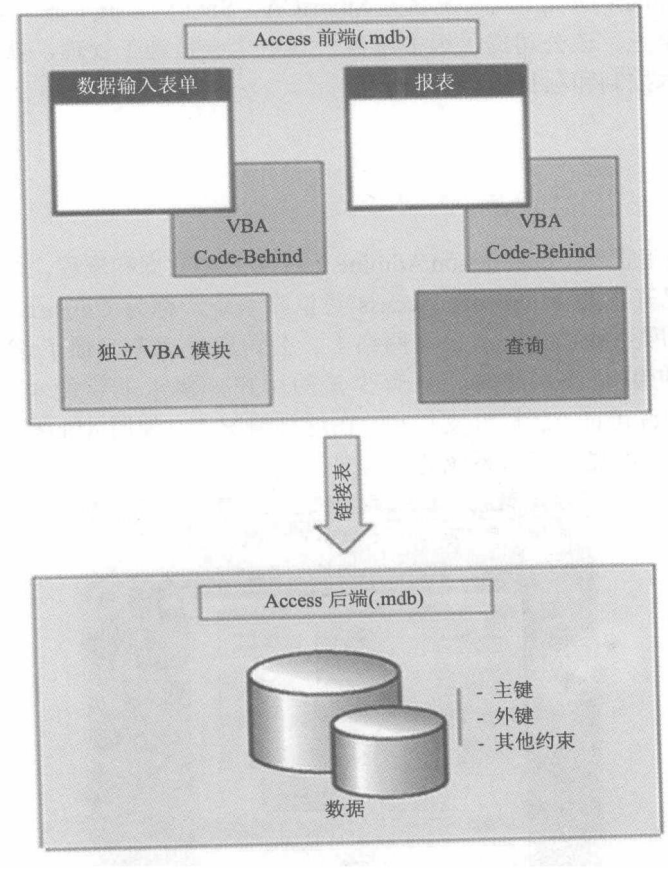


图 1-2　遗留的施工管理系统架构

最近，Smart Design 和另一家建筑设计公司合并，CA 系统变得更加重要。现在，越来越多的用户从几个远程办公室更频繁地使用它，从而导致了伸缩性和性能问题。

Access 应用的问题是，如此多的逻辑嵌进表单、查询和报表里，从而使系统维护和给系统添加新功能非常困难。

1.2　设计

作为一个 Microsoft Access 应用，原来的架构并不差。如图 1-2 所示，它是一个两层的系统，所有 UI、业务逻辑和查询都在第一层(一个分离的.mdb 文件)，所有数据库表在第二层(也是一个分离的.mdb 文件)。虽然当前的施工管理解决方案在 10 年前很合适，但 Smart Design 成长过快，需要一个新的解决方案来支撑新的需要。10 年前，他们的需要相当简单——只需要一种有组织的方式来获取信息，打印报表，发送给不同项目的不同人员。

最初，主要的需求是一个非常简单的信息跟踪和报表工具。系统的第一个版本没有涉及到什么 IT 技术，只有一个涉众和一个 Access 程序员。这些年来，涉众和 Access 程序员都对它做了很多修改。一些修改导致数据结构非规范化、重复代码和各种其他的“代码臭味”。甚至 Access 程序员也经常察觉不到对系统所作的修改，并且修正代码花费了大量的时间和精力，因此，该系统的发展举步维艰。一旦要跟踪的数据开始变多，就使用归档的解决方案，结果是创建了更多的 Microsoft Access 数据文件。最后，几乎每一个试图增强系统所作的改进，都导致某些类型的“一次性”解决方案，日积月累，这样的解决方案变得很难维护。

现在，CA 系统已经被 Smart Design 公司管理层看作公司业务的关键部分，很明显，它已经超出了最初的设计。Smart Design 决定他们不购买现货产品，而是在一个不同平台上重写当前的系统，以满足日益增长的需要。

以下是他们最主要的需要，按重要性排序：

- 可靠性和可得性
- 伸缩性
- 可维护性
- 富客户应用功能
- 离线可得
- Web 访问
- 智能安装和自动更新功能
- 附加的客户设备支持

1.2.1　可靠性和可得性

当前系统的一个问题是，数据库有时会损坏，必须定期压缩或修复，这样，过一段时间，系统就要宕机一次。新系统的数据库应该能够在使用过程中作备份，而且不应该轻易发生数据错乱的问题。

1.2.2 伸缩性

系统应该能够处理日益增长的用户和流程的需要。

1.2.3 可维护性

因为系统的代码没有正确分割，更新当前系统有时就需要在若干不同的位置更新同样的代码和逻辑。新系统的设计必须做到：领域逻辑处于中心位置，而且绝不会重复。

1.2.4 富客户应用功能

用户已经习惯于当前 Microsoft Access 应用的丰富控件和响应性，他们想继续提升这种用户体验。

1.2.5 离线可得

用户没有连接到网络时，新系统必须能够工作。那些偶尔或间歇性使用网络连接的用户——如现场建设经理，并不能随时保证有网络连接——这时，断接时可以工作非常重要。

1.2.6 Web 访问

公司希望系统的一部分暴露在 Web 上，如报表。还有，将来系统的某些部分可能需要暴露给外部承包商。另一个希望是有能力扩展系统，以支持某些类型的管理面板，以向管理层展示关键绩效指标(KPI)或类似信息。

1.2.7 智能安装和自动更新功能

当前，Smart Design 的 IT 部门面临挑战，要确保系统的用户在他们的桌面上有正确的版本。IT 部门也比较头痛，如何在系统有新的变更时推陈出新。IT 部门明确地倾向于类似 Web 应用的部署方法，希望 SmartCA 通过单击内部网的一个 URL 就可以轻松安装。系统必须能够在运行时更新，而且更新应该保证应用及其相关文件的完整性。

1.2.8 附加客户设备支持

新系统应该设计成能够为不同的 UI 设备(如个人数字助理(PDA)、智能手机等)复用大部分核心逻辑模块。

当前系统和平台不太容易支持这些需求。因此，Smart Design 决定从头开始，完全重写一个新系统，以满足新的需求。老的 Access 应用已经为公司服务了 10 年有余。它依然可以作为新设计的基础，继续为公司服务。在老系统中，可以捕获到许多业务规则，这些规则在别的地方并没有归档，因此，老系统将作为一份指南，充实新系统的需求。

1.3　解决方案

新系统 SmartCA，将使用 Microsoft Visual Studio 2008 (包括 Microsoft .NET Framework 3.5)技术编写，包括客户端和服务器端。

1.3.1　满足可靠性、可得性、伸缩性、离线可得和附加客户设备支持需求

大多数当前问题是在可靠性、可得性和伸缩性方面，遗留系统用 Microsoft Access 来实现，并用 Access 作为数据存储。在新的解决方案中，服务器和客户上都将使用一个数据库。

1. 在服务器上

为了支持可靠性和可得性需求，数据库服务器将使用 SQL Server 实例。所有来自遗留系统的数据需要迁移到新的 SQL Server 数据库。为了简化这个数据传输，需要写一段 SQL 迁移脚本或.NET 程序。这样，在构建新系统时老系统可以继续工作，使用脚本或迁移工具，可以更容易定期地从 Access 数据库刷新数据，用来开发、测试和搭建数据库环境。迁移到基于服务器的关系数据库(SQL Server)也同样能更好地满足伸缩性需求，但是系统设计方面还有很多事情要做，不单单是用一个数据库服务器代替 Access.mdb 文件作数据存储这样的想法就行了。

2. 在客户上

是的，您没看错。在客户上有一个数据库。您可能在对自己说，“这比原来的 Access 应用的两层架构还糟糕，至少那个数据库可以在网络上共享！”别着急，朋友。系统的其中一个需求是能够支持并非一直联网的用户，如那些建筑经理，可能呆在建筑拖车里面，没法上网，这就是离线可得需求。在客户上所使用的数据库将会是一个 SQL Server Compact Edition 3.5(SQL CE)数据库。虽然 SQL CE 最开始只是针对移动平台，如 PDA 和 Tablet PC，但是 SQL CE 现在可以运行在所有客户平台上。根据 Microsoft 的定义，SQL CE 是一个“易维护、小巧的嵌入式数据库，针对所有 Windows 平台上的单用户客户应用，包括 tablet PC、pocket PC、智能手机和台式机。和 SQL Server Mobile 一样，SQL Server Compact 是一个免费、易用、轻量和可嵌入的 SQL Server 2005 版本，用来开发桌面和移动应用。”

在客户上有数据库的另一个好处是，它可以帮助分担数据库服务器的一些负载，这样就有助于满足伸缩性需求。

这个时候，您可能会问自己，“为什么不使用 SQL Server Express？至少用 SQL Server Express 我可以使用存储过程！”确实，SQL Server Express 支持存储过程，而 SQL CE 不支持，使用 SQL CE 的真正原因是我想要支持多种设备，而不只是 Windows 机器。使用 SQL