Broadview
www.broadview.com.cn

国际技术期刊中文精华合集

.NET & XML Magazine

2002-2003 中文精华合集 [第1辑]

Fawcette Technical Publications 独家授权

国际技术期刊合集编委会 编译

- 国际名刊精华
- .NET和XML技术荟萃
- Web Services开发必读

電子工業出版社
PUBLISHING HOUSE OF ELECTRONIC INDUSTRY
http://www.phei.com.cn

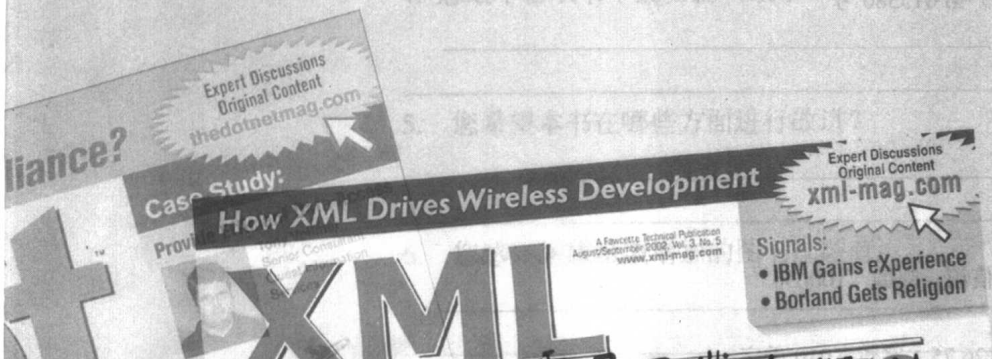
国际技术期刊中文精华合集

.NET & XML Magazine

2002—2003 中文精华合集 [第1辑]

Fawcette Technical Publications 独家授权

国际技术期刊合集编委会 编译



電子工業出版社
Publishing House of Electronics Industry
北京 • BEIJING

内 容 简 介

本书汇集了国际著名技术媒体 Fawcette Technical Publications 旗下技术期刊.NET Magazine 和 XML Web Services Magazine 2002—2003 年度精华文章数十篇。主要涉及.NET 平台和 Web Services 技术平台，由各自领域内的一流专家撰写，其内容包括了从编程技术到产品配置，从工具使用技巧到新技术剖析的各个方面，技术含量丰富，观点权威，涵盖面广。

本书不仅适合专业软件开发者阅读学习和参考，同时也适合广大技术爱好者、在校学生和教师阅读学习。

版权贸易合同登记号　图字：01-2003-7710

图书在版编目（CIP）数据

.NET & XML Magazine 2002—2003 中文精华合集. 第 1 辑 / 美国发赛特技术出版集团著；国际技术期刊合集编委会编译. —北京：电子工业出版社，2004.4
（国际技术期刊中文精华合集）
书名原文：.NET Magazine & XML Web Service Magazine
ISBN 7-5053-9718-4

Ⅰ.N...　Ⅱ.①美...②国...　Ⅲ.①计算机网络—程序设计—文集②可扩充语言，XML—程序设计—文集
Ⅳ.①TP393-53　②TP312-53

中国版本图书馆 CIP 数据核字（2004）第 015380 号

责任编辑：周　筠　方　舟
印　　刷：北京中科印刷有限公司
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编　100036
经　　销：各地新华书店
开　　本：880×1230　1/16　印张：26.75　字数：910 千字
印　　次：2004 年 4 月第 1 次印刷
印　　数：6000 册　　定价：38.00 元

凡购买电子工业出版社的图书，如有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系。联系电话：（010）68279077。质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

前　言

这是一本开发技术文集，收录了国际著名开发技术期刊.NET Magazine 和 XML Web Services Magazine 杂志 2002—2003 年发表的一批精华文章。

程序员作为一个职业群体，不但需要系统学习基础和经典理论，在实际项目中不断磨练自己，而且需要与同行充分交流，保持对新技术和新思想的敏感，向同行中的佼佼者取经。在这一点上，技术期刊能够起到别的载体无法替代的作用，而国外的高水平技术期刊，在这方面更是有着辉煌的传统。技术期刊中的文章，一方面能紧跟最新的技术趋势，另一方面来自名家之作，高屋建瓴，能够反映比较高的技术水平和崭新的视角。在 IT 产业发展的历史上，技术期刊起到了重要的推动作用。很多重要的新技术和新思想，最初都是发表在期刊上，进而对整个产业界产生深刻影响。因此，作为一名不断追求进步的程序员，应当经常地阅读高水平的国际技术期刊。

Fawcette Technical Publications（发赛特出版集团，简称 FTP 集团）是美国著名的 IT 技术媒体和出版集团，旗下拥有著名的 VSLive!技术大会以及 Visual Studio Magazine、Java Pro Magazine、.NET Magazine 和 XML Web Services Magazine 等技术期刊。FTP 集团的技术期刊拥有一大批世界知名的技术作家，如 VB 之父 Alan Cooper、.NET 专家 Dino Esposito、Java 专家 James Cooper 等，一向以文章质量高、观点权威新颖、技术含量丰富著称。

自从微软公司在新千年向世界公布.NET 战略以来，面对来自 Sun 公司 Java 平台的巨大挑战和技术更新所带来的巨大压力，.NET 技术在短短几年时间中迅速发展，逐渐被工业界、教育界和学术界所接受，成为与抢占先机的 Java 平台技术并肩、最主流的软件开发、技术研究和企业应用平台之一。而.NET 平台所仰赖的核心理念之中，所谓 Web Services 概念和前景无限的 XML 技术无疑是.NET 最重的两个砝码。目前，.NET 在国内应用日渐广泛，广大.NET 开发者急需通过阅读高水平、国际化的.NET 技术刊物迅速提升自己的技术水平。

.NET Magazine 自 2001 年创刊，XML Web Services Magazine 自 1999 年创刊。几年来，这两本期刊在.NET 的脚步下一路发展壮大，在国际.NET 技术界享有盛誉。此次我们本着为广大专业开发人员和学习者服务的宗旨，引进.NET Magazine 和 XML Web Services Magazine 杂志，将其精华内容集结成书，翻译出版。在选编时，我们尽可能注意技术期刊内容的特点，既选择一些最新、最前沿的技术文章，也选择一些立意深远、具有长期意义的文章。希望这本精华合集既能够满足开发者平时学习工作的要求，也能够具有一定的保存价值。这是我们的一个尝试，希望得到广大读者的积极反馈。

全书共 19 篇，分别收录了.NET Magazine 和 XML Web Services Magazine 2002 年 3 月—2003 年 2 月各期的内容，以及 2003 年 3—9 月的文章精选。

本书由“国际技术期刊合集编委会”主持编译，由赵辉、谢益煌和王伟宏进行选编和翻译。

国际技术期刊的选、编、译，对于我们来说是一个崭新的尝试，限于经验和水平，失误之处在所难免，恳请读者不吝赐教。

国际技术期刊合集编委会

2004 年 2 月

目 录

.NET & XML


Expert Discussions
Original Content
thedotnetmag.com
How XML Drives Wireless Development
XML
& WEB SERVICES
MAGAZINE
第一部分
.NET
Flowcharting the Future of
Application
Architecture
Service
Magazine

.NET & XML

第一篇

- 为.NET 调整你的 Web 架构
- 克服采纳.NET 的商业障碍
- 定位 Web 服务

为.NET 调整你的 web 架构

当你转向使用.NET 时，你可能会惊奇地发现，通过重新审查你的多层架构，你的生意从中收益是多么大啊。

● Barry Bloom

技术工具箱： Visual Studio .NET，ASP.NET，.NET 框架

没有什么能比在微软的网站上阅读关于服务器和应用程序架构方面的科学文章更让我为之疯狂了。当你以多层架构为内容做一个查找时，你将会得到一个庞大的结果集。目前有很多来自微软各个开发团体的关于这方面的文章，使得它们想表达的意思有些混乱。在这篇文章中，我将会为你解释为什么多层架构的作用被夸大了，以及.NET 是如何改变你对多层架构的认识；我也会探讨一下如何在企业内部因特网的环境下使用.NET Web 服务，以及.NET 框架是如何影响 Web 应用程序本身的。

我认为严密构思应用程序的设计和为分布式 Web 应用程序建立一些设计原则和方针是非常重要的。我阅读和撰写了一些有关多层架构的文章，并实现了多层架构以及通过无状态的中间层组件成功地获取了可扩展性。但最近当我读到或者听到关于多层架构设计的重要性时，我感觉不是很好。不要误解我的意思，我并不是要去批判多层架构的基本原理，我也不是说它一点价值都没有，因为它的确是有好处的。然而，我觉得微软和其他公司对多层架构这一概念的吹捧远超过了它的实际使用的范围。全世界数以千计的 IT 行业的从业人士都简单地认为建造多层架构是理所当然的，而不需要其他理由。

尽管在微软公司没人会跟你说你应当放弃多层架构（至少我没听过），但是如果回顾一下在事物处理性能委员会（TPC）关于 Web 应用程序制定的评测中的大赢家，你会发现那些基本的应用程序的代码几乎跟多层架构没什么关系。为了赢得这些测试，微软的开发者实现了一种用基于 C 语言的因特网服务器的 API（ISAPI）的动态链接库，用来处理所有的网络请求并在后台管理所有对数据库的访问动作，这些代码全都是运行在网络服务器进程中的表示层（Web-tier）。这样就找不到中间层了。

微软的确对多层架构的很多方案做了很多比较测试，而且性能也很不错。但没什么方案可以跟这种“不惜任何代价地只为了性能”的方案相匹敌，我把它称为整体式的 Web 应用程序。进行这种测试所面临的问题在于，没有一个公司可以实现满足微软测试所要求的条件的网站。它所需要的时间太多，需要的技能是大多数网络开发者所不具备、难以改变和管理的，并且跟 RAD 相差太远。我相信微软在这些比较测试中的成功，连同人们的常识会改变人们对.NET 平台上 Web 应用程序架构上的看法。

.NET 框架和 ASP.NET 都提倡小规模多层架构的方案，并提供了 ASP/Visual Basic 多层 Web 应用程序的 RAD 能力，相信这不只是巧合。.NET 框架吸收了成功基准的原则，并将其与 VS.NET 中的 RAD 工具结合起来，从而实现了两方面的性能。例如，在 COM 的世界里，如果想访问数据库，可以写一个无状态的组件，让它在 COM+中运行。随着每次使用，以及根据请求从数据库中取出数据，这个组件会被创建和销毁各一次。如果按照多层架构的服务器模型，这些组件潜在地运行在不同服务器 ASP 网页上，据推测，这个进程会为应用程序的扩展提供方便。但我从未清楚理解这是怎样实现的，特别是对于静态数据来说。聪明的开发人员用分离的记录集来缓存这些数据，或者使用其他古老的技术，即便是在 COM+的世界中也是如此。但是不应该那样做，而且那也并不容易做到。

COM 过时了，但无关紧要

在.NET 中，我们使用网络服务器上的一个数据集来访问数据库，并将其保存在缓存中以便于下次使用。在初

始化时从数据库中调用，然后你就可以在整个应用程序的生存周期里使用缓存中的数据。在.NET 中，COM+类型的中间层数据访问形式不存在了，而是被移到各自的网络服务器的后端（如图 1 所示）。

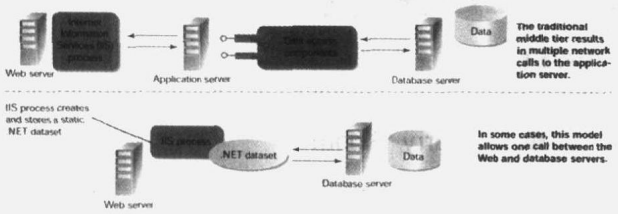


图 1　挑选和选择

这一改变并不意味着开发人员要回退到混合外观代码和数据访问代码的程度。如果你还沉醉在 20 世纪 90 年代初期客户端/服务器的时代，那你就真的需要一台时间机器来让你回到过去那个混乱的时代。在.NET 中，你还是按照多层架构的方法组织和开发你的代码，但是你不再须要考虑诸如数据存取之类的基本需求所需要的传统的中间层应用程序服务器了。对大多数人来说，这是显而易见的。但是你仍然会为很多人对此持有不同看法而感到惊讶。

那么 COM 就没用了吗？我不这么认为，不过，当微软发行.NET 框架时，它倒是首次遭到重创。如果我真的需要多种资源类型的分布式事务支持，则可能会使用 COM+，但一般而言，我觉得 COM+即将退出历史舞台。这对包括我在内很多自认为是 COM 方面专家的人来说，是难以下咽的苦果。如果说.NET 是建立在 COM 的基础上的，可能会使你好过一点；这一点被很好地隐藏了，我们再也不用去提及它。当然，如果 COM 过时了，那分布式组件对象模型（DCOM）会怎样呢？

Web 服务：下一代的 DCOM

在我的公司，经过测试并仔细研究，我们创建了一些企业内部因特网的 Web 服务来代替现有的 DCOM 服务。其他应用程序使用这些企业内部因特网 Web 服务，就像你用一个数据库一样。这同微软的设想存在一些不同，微软的目的是服务器之间的大多数 Web 服务的访问通过因特网实现（如图 2 所示）。

在应用程序架构的内部，用.NET Web 服务取代 DCOM 服务，在微软的社区曾引起过一场很大的争论，因为它背离了最初的设计规范的意图。.NET Web 服务很详

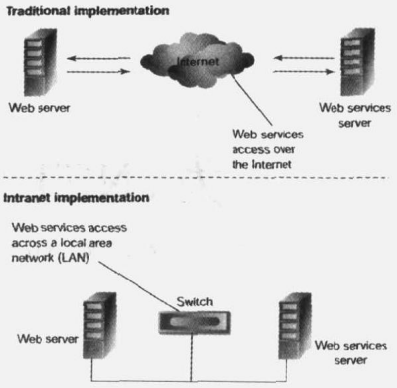


图 2　尝试一些新的东西

细、全面并且基于标准，因为它遵循因特网间的跨服务器的通信的简单对象访问协议（SOAP）的 XML 规范。与之相反，DCOM 是一个最优化的二进制位格式，因为服务器使用 XML 来进行通信，所以传递的信息可以让人读懂，这对应用程序的设计有着重要的含义：高效地支持 Web 服务的 Web 应用程序应当考虑到内部服务器的通信细节，比如调用次数、以及要被转发的 SOAP 数据包的大小。传统上的 DCOM 没有这些问题，所以并不是每一个 DCOM 应用程序都可以很好地“移植”成.NET Web 服务。由于 VS.NET 使得运行.NET Web 服务更加容易，所以我预测到年底为止，在工业界中将会普遍采用类似于 DCOM 的企业内部因特网.NET Web 服务。

当我的公司决定把一些现有的 DCOM 应用程序移植成 Web 服务时，我们探讨了在什么情况下，应用程序中的 DCOM 服务须要重写成.NET Web 服务，并制定了以下 5 项标准：

1．不限于 Windows 的跨平台企业级访问要求

由于 SOAP 是基于 XML 的协议，所以它非常适用于 Web 服务，而且任何平台都可以实现 Web 服务或者跟 Web 服务通信。我们使用一个.NET Web 服务把原有的大型主机与 NT 系统相连接，效果还不错。

2．排除它存在于自身服务器组上的特殊硬件要求

例如，我们的信用卡处理服务器通过“黑盒子”路由器跟银行直接相连。在这一簇中的每一个服务器都需要一块附加的网络接口卡。

3．无法协调网络服务器间实行自动化配置的复杂软件设置和安装。

这通常指的是第三方应用程序，它们缺乏微软服务器产品的部署活力，或者设计是跨平台的。

4．每个服务器许可模式很昂贵，而且跨 Web server

farm 分布是不实际的

当独立的软件提供商理解了用 Web 服务包装产品是多么容易时，你可以期望将这些产品的大部分移植到一个事务处理许可模式。

5．来自应用程序的单个远程调用

关于客户如何使用这些应用程序，都有一个“快速响应”需求。Web 服务器将做一个远程调用，等待回复，然后继续处理。

通常而言，如果一个 DCOM 服务满足了至少上面五条标准中的三条，我们会考虑通过.NET 应用程序访问它，将它作为企业内部因特网的 Web 服务。我们进行了载量的测试，证明这些新的 Web Services 与目前使用的 DCOM 应用程序一样，能处理相同的装载量。有一次，我们用 Web 服务获得的每秒事务处理量是原 DCOM 服务的两倍。通过换掉复杂的 COM+应用程序，用简单的 C# Web 服务代替，使调用次数大大减少了，这样我们就可以为更多的用户提供服务，而不增加服务器负担。但是，Web 服务并不是惟一地影响 Web 应用程序架构的.NET 性能。接下来，我将论述新的.NET 框架的设计性能。

线程虽小但作用重大

四年前，当我第一次开始开发 Web 应用程序时，我是在一个应用程序开发公司，那里的每一个人都是 C++开发人员，并且都有丰富的 COM 经验。我习惯于用微软的 API 所集中的所有工具来解决问题。但当我在我的新公司开始工作时，我把 C++的技能和 API 的经验搁到一边，转而学习如何像 ASP 网络开发者一样地思考问题。

然而，在某些时候我会遇到一些问题，并希望能更容易地通过 API 工具来解决问题。我的思考方式使我的解决方案并不像我所期待的那样能够适应网络环境。显然，微软的开发人员经历过同样的困惑，因为他们为像我这样的人创建了.NET 框架。.NET 框架为开发人员提供了所有的微软工具箱里的工具，可以用于 Web 应用程序的开发，他们不再须要像 ASP 网络开发人员那样考虑创建能够用于网络的解决方案。开发人员仍然须要通过构建一个网站来获得经验和技巧，来明确设计时做的选择，但使用.NET 框架，也可以用他们选择的任何方式来处理后端用户需求。不要低估这个新工具会给以 ASP.NET 编写的 Web 应用程序带来的重要性。

可以预见的是，明年或者晚些时候，我们会看到许多现有的 ASP 应用程序直接移植到 ASP.NET。因为后者的内置缓存技术和编译执行方式能为 ASP 代码的运行带来性能的提高，然后开发者将能从.NET 框架的性能中得到补充和支持。我想在一个用户访问一个网站期间，我们最初会看到一个个的异步进程。当今的 ASP 中，一个页面上用来满足用户的偶然请求的所有任务是以线性的方式执行的。如果有两个数据库查询任务，每一个须要花费五秒的时间，这两个任务必须相继地执行，用户必须等待十秒才能得到结果。而 ASP.NET 呢，相同的情况在五秒内就可以完成，因为它是通过创建一个附加的线程来处理另一个请求，达到了两个查询并行处理的目的。

我也注意到我所称呼的 farmer threads 的形成过程。这些 farmer threads 是在应用程序初始化时创建的，并存在于一个特定服务器上的网站生存周期里。它们执行与网站维护和管理相关的任务，检查是否有新内容，很好地进行更新，或者监控一个网站需要的资源。这就保证了外部数据库、邮件服务器等都是可用的，并且运行正常，这是与诸如 Microsoft Operations Manager 传统的代理监控工具不同的。有无数可能完成的任务，而这一切只是用了线程。如果你想启动你的 ASP.NET 开发项目，你可以雇一些传统的 C++ Windows 程序员，让他们同 Web 开发人员一起工作。如果用 C++程序员的能力以及 Web 开发人员的经验，那么你将改变世界！

.NET 平台给 Web 架构带来巨大的影响，任何架构都会衰退，.NET 也一样。理解它的影响，对是否沿用 COM 的概念做出正确的决定，在内部使用 Web 服务，期盼新的.NET 设计能力，你将为公司构造出适合的 Web 架构。

关于作者

Barry Bloom 是 Mary Kay 有限公司电子商务和技术工程的总架构设计师。他负责 Mary Kay 公司的网站的架构，该网站帮助 700,000 妇女经营她们的 Mary Kay 业务。在过去的三年中，他的网站架构使 Mary Kay 75%的订单都从写信订购方式转向网上订购方式，这带来了超过 10 亿美元的零售额。同时，Barry Bloom 也是《Deploying and Managing Microsoft .NET Web Farms》一书的作者（2001 年，SAMS 出版）。

他的电子邮件是：barrydbloom@hotmail.com。

克服采纳.NET的商业障碍

描绘在.NET的迁移过程中涉及到的花费和活动时间表的一幅清晰的图画，为你的团队建立一个正确的计划。

● Brian Noyes

技术工具箱： Visual Studio .NET

如果你的公司正在开发 Windows 上运行的 Web 应用或者产品，那么，实际上可肯定的是，你将在某个时候向.NET 迁移。你读这本杂志这个简单事实也说明，你正在考虑迁移到.NET。但是在你启动迁移计划之前，你须回答两个重要的问题：应该在什么时候迁移？在什么地方花费钱和精力？这两个问题的答案在很大程度上并不固定——基于你当前开发的工程、公司规模、员工的专门技能以及很多其他因素——因此我不能给出简单答案来适合每种情况。但是，我能讨论各种因素，它们是你决定何时实施迁移和迁移投资规模时必须考虑的。

采纳.NET 将意味着不同的事物对应不同的公司。最小规模的迁移会简单地将你的开发环境升级到 Visual Studio .NET (VS.NET)，但与此同时，继续进行非.NET 应用的开发。最大规模的迁移可以在 VS.NET 下启动一个全新的工程，使用.NET 的语言和.NET 的框架类，把应用建立在.NET 企业服务器基础上。成本、培训、软件工程的类型和时间等因素都会影响你的组织正确的发展方向。另外也须要意识到，这是一个全新开发平台的第一个版本，所以在采纳.NET 的初期将伴随着一些危险。

成本估计

当迁移进入.NET 时，有一些费用因素需要我们去评估。你须要计算迁移到.NET 的购买过程和实现过程的大致成本和迁移到 VS.NET 的购买过程和实现过程的精确成本；而且估计在无形产品的获取和维护上面你能够节省的数目（如图 1 所示）。对于一家大公司，实现成本往往比购买 VS.NET 的成本要高出很多。然而，对于一家小公司，需要慎重考虑购买成本，它甚至成为公司快速迁移的障碍。

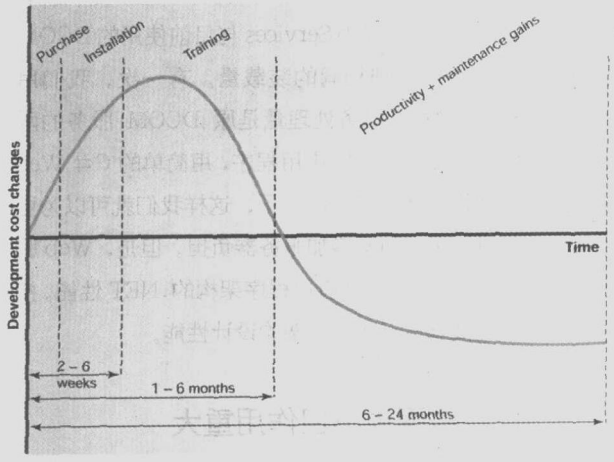


图 1　开发成本的变化

.NET 迁移的初期成本并不是无关紧要的。培训和短期的生产损失将占用大部分的花费。但是由于生产的收入和减少的维护成本所带来的长期利润应该大大超过短期的开销。

虽然在采纳.NET 技术时，你的公司实际上只能够使用软件开发工具箱 (Software Development Kit (SDK))，而不用使用 VS.NET，但从近期来看这样的决策不是很实际。第三方开发工具市场的繁荣以及真正开始提供可以与 VS.NET 竞争的工具还需要一段时间，在此之前，仅仅为了节省购买VS.NET的费用而让你的开发人员使用SDK里面的命令行工具是得不偿失的。

当你评价成本的时候，计算一下潜在的对其他你已经购买或者计划去购买的开发工具在成本上的影响。例如，你想购买设计和建模工具，注意到包括在 VS.NET 企业版内的 VS.NET Enterprise Architect (VS.NET 企业架构师) 将与一个专业版本的矢量绘图软件 (Visio) 一起装运，Visio 具备了可与其他专用工具媲美的完整 UML 建模功能。Enterprise Architect 也包括企业的管理、分析、报表的产

生和能影响你购买其他工具的其他工具。因此，如果你正要购买一个报告产生工具，或许应该先看看包含在 Enterprise Architect 中的 Crystal Reports 的功能是否能够满足要求。

也评价一下你的开发者目前在 Visual Studio 环境内使用的工具。保证他们仍然在新的集成开发环境（Integrated Development Environment(IDE)）里工作，或至少他们的制造商在不远的将来计划更新并将那些费用加入预算中。

对于很多公司，VS.NET 工具的价格与实现成本相比显得微不足道。购买产品之后，你须要计算在没有打乱正常工作的情况下，在你的开发者的机器上安装新产品和 SDK 的费用；考虑到你的硬件可能须要做一些升级工作使 VS.NET 流畅地运行。迁移总成本中另一个重要的组成部分可能是对开发者、测试者和其他 IT 职员的培训；计算出他们须要多少正式和非正式的培训，这些培训基于你迁移的快慢和范围。此外，在那些过渡时期和在你员工加快速度之后，都要考虑实现附加的报告来跟踪生产力的变化并且显示投资回报率（ROI）。对这些方面进行审计的全面和深入程度也将影响到实现成本。

如果你是一个正在进行精简的组织，那么你可以支付迁移的费用——与此同时提高你的公司的能力——依靠将一些费用分担到你的客户的方式。你将须要提出正确的证据使他们确信在他们的系统中追求.NET 是他们最大的利益，因为你将能够提供更好、更迅速和更便宜的附加性能。如果成功的话，你将能分配一些你在迁移中的一些成本。

跨越.NET 的知识障碍

在评估成本后，要决定怎样使你的人员向.NET 的迁移做准备。.NET 开发技术有一条急剧升降的学习曲线，它的变化相当大地取决于你的开发者的程序设计经验。VB 开发者有一条最陡的曲线，因为它们须要适应在句法上的变化并且采用面向对象（OO）的方法学。C++开发者能保持发展既有编码，但是他们也将容易地适应于 managed C++或者 C#。Web 开发者将须要适应 OO 程序设计，如果他们想要开始利用 ASP.NET 的类来实现 Web Form 和 Web 服务。但是，你的全部开发者现在将能够协作并且分享知识，因为全部语言分享在.NET 架构内的相同的类库。你的开发团队应该发现 VS.NET 环境从 VS6 产品起有一个大的改进。

如果你能使一些或者全部开发者都集中在.NET 培训中，则将增加成功迁移的机会。明智地选择培训，为期一周的新兵培训形式是最好的方法之一。保证培训者已经使用.NET 有至少 6 个月的时间，检查看看他们是否能使会议聚焦在你的公司发展的类型上。.NET 是一只多头和无定形的野兽，集中于 Web 服务和 ASP.NET 的.NET 培训不会给你的团队带来很多好处，如果他们建造的是普通的应用软件。如果你不能或者不想要派你的全部人员都参与正式培训，那么就派一些关键成员，然后建立一个内部培训计划，使你的整个.NET 团队一起分享他们的知识。

在.NET 中抓住你的工程

进行.NET 迁移规划时，还有一个必须考虑的问题，它也会对最终成本和培训需求产生重要影响，这就是如何调整现有的工程使其适应.NET 环境。很少公司允许他们从零做起，重新开发.NET 应用程序。更加有可能的是，你必须与现有产品和系统一起评估兼容性问题。如果你开始了一个新合同或者新工程，那么使用最干净的方法并且完全使用.NET 工具和技术开发它。但是一些工程还须要使用一些你不能抛弃的现有的类库、组件或者其他代码。要解决这些问题，写出一个计划，说明你将如何迁移、结合，或者放弃你现有的代码和工程（如图 2 所示）。

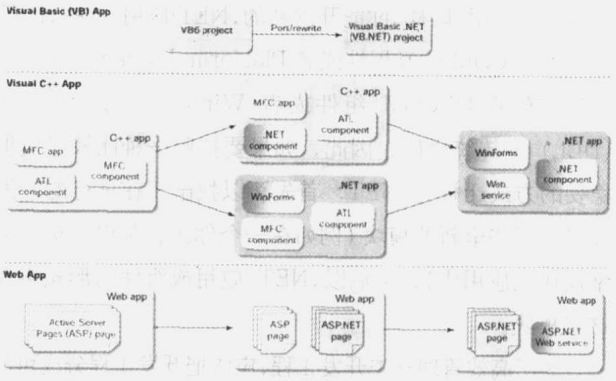


图 2　挑选你的路径

如果你的工程是用 VB 6 编写，你将面临一些艰苦的决定。其中一个选择是完全将代码转换成.NET，第二个是继续使用 VB 6。但是，VB 6 的代码不与 VB.NET 的编译器和类库相兼容。迁移工具能帮助代码的转换，但是你仍然须做大量的工作将一个复杂工程完全地迁移到.NET。

.NET 开发的路径取决于你须迁移到什么样的工程。Visual Basic 6（VB 6）将是一个完整的端口，而 C++应

用程序可以在整体迁移之前利用互操作性功能迁移其中的一部分。Web 应用程序可以一起运行 Active Server Pages (ASP)和 ASP.NET，然后逐渐迁移到.NET。

一个包含了大部分 C++代码的工程由一些 MFC 类(微软基础类 Microsoft Foundation Classes)、ActiveX 模板库 (ActiveX Template Library(ATL)) 和 Win32 API 调用的结合组成，它需要我们下一些另外的决定。首先，转到 VS.NET 能够有效地提高开发效率，而且也为利用.NET 开发工程的某些部分做好了准备（你能混合 VS6 和 VS.NET 开发相同的工程，但是针对旧的和新的 IDE，你必须在项目计划文件中复制并且同步配置信息——这不是一个好的解决办法）；其次，在.NET 内发展新的 GUI（图形用户界面）或者组件，但仍通过.NET 平台提供的互操作功能利用原来的代码。

在逐渐迁移的过程中，Web 应用提出最容易的方案。如果你有一个动态服务器主页（Active Server Pages (ASP)）工程，那么开始引进与 ASP 网页一起运行的 ASP.NET 网页。虽然这些网页在单独的进程里运行，但是它们对终端用户是透明的。因此，你将须要使用一数据库或者其他持久的存储空间，在这些网页中共享公共的状态。

对于任何方案来说，你能先使用.NET 发展新的组件，然后利用在.NET 框架内的 COM 的互操作特性，把它们并入传统应用。同样，你能开发新的.NET 应用；这些.NET 应用通过 COM 的互用性或者 Platform Invoke 功能，使用现有的作为 COM 组件或者 Windows 动态链接库 (DLL) 的传统代码。因此，如果要按照一种比较缓慢但稳妥的方式迁移到.NET，首先可以找出正在进行的开发项目中那些全新的模块（例如在一个你正在加以改进的大型模块式应用中），然后以.NET 应用或组件的形式实现这些模块。

为了高效管理这类开发工程，应该把开发工程分成可分别独立开发的传统代码部分和.NET 代码部分，利用组件开发技术定义各个部分都应该遵从的接口。然后，如果可能的话，就用.NET 开发组件；如果要面对大量不可能一下子全部移植到.NET 的传统代码，则仍使用 VB6、ATL 或 MFC。

时间就是一切

规划好.NET 迁移计划中成本、培训、工程管理方面的因素之后，接下来要考虑的显然就是引入.NET 技术的合适时机。如果你的开发组正在开发一组 ATL 组件，而且有一部分的工作已经完成，那么，仅仅因为.NET 的发行而让开发组改用.NET 从头设计组件是不可取的；但是，如果这些正在开发的组件属于一个大型.NET 开发项目最前期的一小部分，中途改换开发平台也许是值得的。

如前所述，对于新的开发项目，从一开始就完全采用.NET 技术是最简单和直接的途径。但是，如果开发项目中已完成的工作占了较大的比重，或要对一个大规模的系统进行维护和扩展，决定何时引入.NET 技术就要困难多了。如果软件系统或产品的生命已经接近尽头，那就不必自找麻烦，因为代码互操作带来的复杂性、培训和性能影响等方面的代价，很可能超过少量.NET 代码带来的优势。但是，如果预期软件项目还有数年以上的生存时间，努力把整个系统逐步迁移到.NET 可能更为合理。具体的做法是以.NET 组件或应用的形式分阶段引入新的模块。

迁移的时间计划中还应该考虑到职员的稳定性问题。如果打算对职员安排做重大的调整，在实施调整之前，首先应该明确.NET 迁移策略。传统技术或.NET 技术的选择将对职员类型和技能要求产生重大影响。例如，传统的应用要求熟悉脚本编程的程序员，而.NET Web 应用要求具有 Web 开发经验且熟悉 OO 编程的程序员。如果你有一些传统的 C++应用，就不应该雇用只懂 VB 的程序员来维护代码。然而，对于.NET 开发，VB 程序员和 C++以及 C#程序员可以并肩工作，因为他们都使用.NET 框架下同样的编程模式和库。也就是说，在.NET 下，不同之处只是语法，而不是以前在编程模式上的截然不同。

迁移到.NET 是一件举足轻重的事情，它暗示着短期之内的多项重大投入。但是，许多研究.NET 的业界分析人士和开发者得到的结论都一样：从长远的眼光来看，生产效率、性能、维护简易性等方面的优势证明对.NET 迁移的投资是值得的。

关于作者

Brian Noyes 是 Digital Access Corporation 公司的一位软件工程经理。他是一位微软认证解决方案开发人员 (MCSD)，在软件工程和系统工程方面有超过十年的经验。他是 Visual Studio 杂志和.NET 杂志的一个技术编辑，他对这两份杂志做出了很大的贡献。他的 E-mail 地址是：bnoyes@domeworks.com。