从 Windows DNA 到.NET

作者： 张晓鹏

Tuesday, August 13 2002 3:29 PM

简介：面对微软有史以来最大规模的技术更新，许许多多挂着.NET **称号的技术和产品让许多人在开始学习时如坠云雾里。但我们知道**.NET **并不是无缘无故从天而降的，回忆一下微软技术的演变，也许能帮我们更加清晰自然的理解**.NET **的内涵。这篇文章重点介绍了微软以前的框架和组件技术，并详细说明了**.NET **对于** Windows DNA **的演进，希望让读者可以更好的理解现在的**.NET**。**

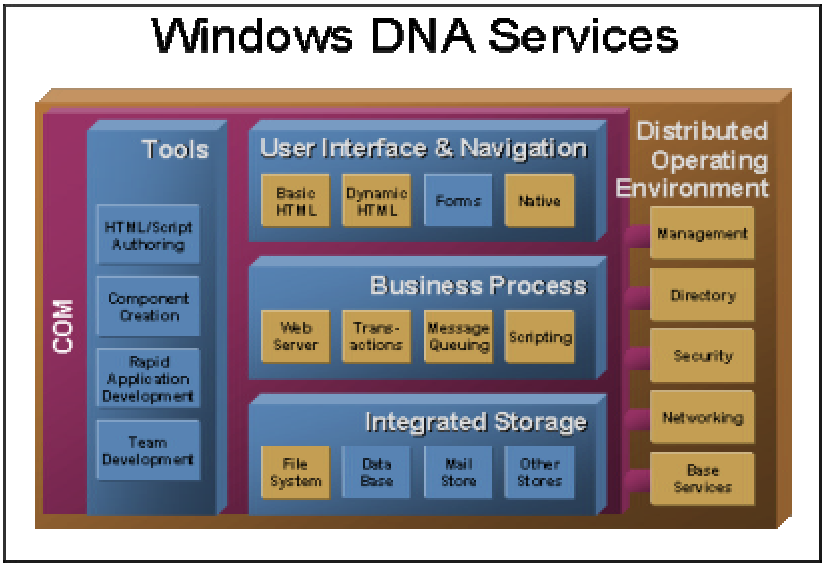
一、Windows DNA

Windows DNA **是** Windows Distributed Internet Application Architecture**的缩写，可以翻译为** Windows **分布式网络应用程序体系结构，这是一个相当抽象的概念。但它又是非常重要的一个概念。微软提出的** DNA **概念是借助生命科学中脱氧核糖核酸（**DNA,Deoxyribonucleic Acid**）的寓意来诠释现代企业信息结构的真谛。比尔·盖茨称之为数字神经系统，寓示信息系统可以灵活适应外界环境因素的变化，做出相应的反应。**

Windows DNA **是过去在微软平台上进行技术开发的大环境，要利用微软的组件技术** OLE、COM、DCOM、MTS、COM+**进行开发，就不能不了解这个** Windows **环境下的软件体系结构谈起，只有了解了这个大环境，我们才能够知道为什么会有这些技术，它们都有哪些作用。**

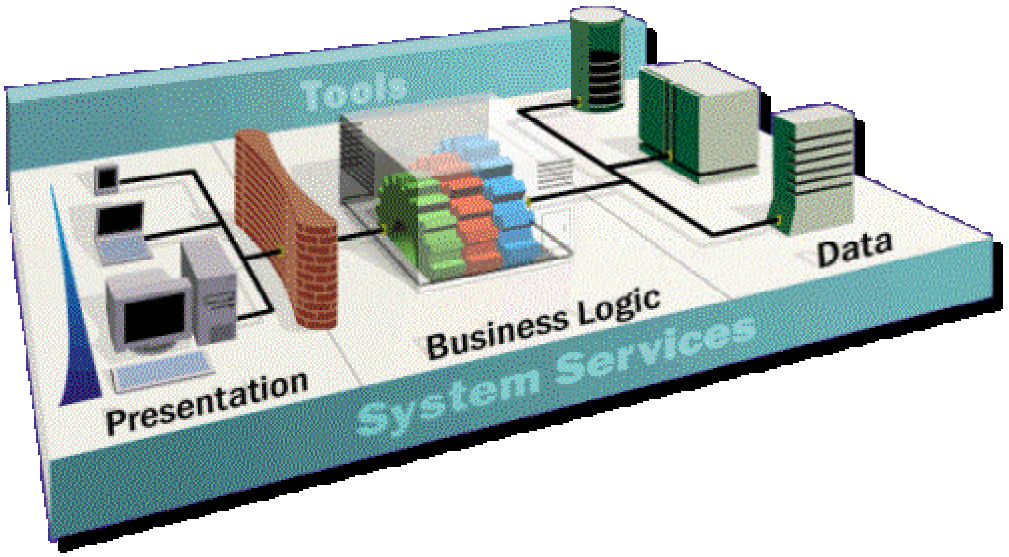
在过去 20 **年中，我们的生活中出现了两种极为重要的技术，一种是今天家喻户晓的互联网** Internet**，另外一个是成本很低，但是功能极为强大的** pc **机。这两种技术在过去都是并行发展的，并且在某种程度上是相互促进的。但是它们对于彼此的支持却不充分，它们都没有充分利用彼此巨大的能量。微软发展** Windows DNA **的目的就是为在** Windows **平台上的应用开发提供一个框架和环境，整合个人电脑和** Internet **的优势。在最高层次上，**Windows DNA **允许不同网络的计算机互相操作以及相互协作以完成某些目标，它可以使开发者很容易的建造能够服务许多用户的基于网络的系统。更为重要的是，**Windows DNA **提供了一个具备协同工作能力的框架（**Framework**），而且由于这个框架支持公用的协议，以及它发布了一些通用的接口，用户可以在它上面添加一些新的功能以扩充这个系统。这也意味着** Windows DNA**提供了一个钩子（**hooks**），第三方可以在** Windows DNA **的基础上添加他们自己的产品，以扩展** Windows DNA **的系统架构。**

Windows DNA **使用了一系列的服务来完成它的架构。例如它使用了组件（**Components**）、**DHTML、WEB **浏览器（**IE**）、**WEB **服务器（**IIS**）、事务管理、消息队列、安全机制、系统管理、用户界面、数据库存取等等。微软扩充的** Windows DNA **包含了工具、数据库、操作系统、编程模型和开发者为企业建立应用程序所需要的应用程序服务。例如** Microsoft Windows 2000 **和** COM +**，是** Windows DNA 2000 **的一部分的，**Microsoft Visual Studio **等开发工具和** Microsoft SQL Server **等数据库也是。其他工具集在可用时，也会成为** Windows DNA 2000 **的一部分。当开发人员遵循** Windows DNA **来开发时，就会半被迫的采用基于构件的开发方式，这是** Windows DNA**强力推行，并且它自己也实践了的。另外采用** Windows DNA **的结构相当于它为开发人员做了大量的安全管理、事务管理、数据库存取等基础服务工作，从而让开发人员集中精力开发有意义的业务逻辑部分。下面我们看看微软是如何把所有这些东西都整合在** Windows DNA **的系统架构里面的。我们用下面这张图来说明：**

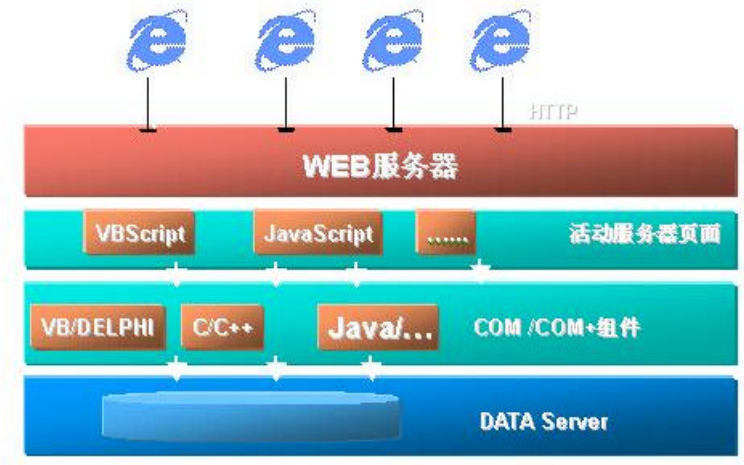


Windows DNA **是微软为分布式应用程序的开发所提供的平台，它是微软将多年的技术精华集合起来而形成一个完整的多层结构的企业应用总体方案，它使** Windows **真正成为企业应用平台。** Windows DNA **实际上是微软的基于组件的分布式应用程序战略框架结构，具有可伸缩性和高可靠性。**

在 Windows DNA **下，对比于上图，开发模型可以入下图所示：**



从这个模型我们可以引申出很多模型来，例如对于基于 WEB **的应用程序，可以用下面的模型来表示：**



二、微软组件技术的演变--从 COM 到 DCOM 和 MTS，直到 COM＋

自从 80 年代面向对象的思想出现以来，从面向对象的程序设计语言到面向对象的程序设计方法，面向对象的思想迅速发展，渗透到计算机科学的每一个领域中来。在过去的 10 年中，对象技术的发展路径是从 DDE（动态数据交换）、OLE（对象链接和镶入）、COM（组件对象模型）和 ActiveX。在这些技术的发展过程中，始终遵循着一个思想：复用，使应用程序开发的复杂性降低。

但是 Internet/Intranet 的飞速发展对应用软件提出了更高的要求，使得对象这个概念有些"小"，已经不适合 Internet/Intranet 下的分布式软件模型，于是组件化程序设计的思想得到了迅速发展。按照组件化程序设计的思想，复杂的应用程序被设计成一些小的，功能单一的组件模块，这些组件模块可以运行在同一台机器上，也可以运行在不同机器上。大家可以想想，为了使这些跨越不同机器、不同操作系统的组件能够相互通信，这些组件必须遵循一个统一的标准，为此，微软提出了 COM 标准，同时也给出了这些组件相互协作的环境。COM 定义了组件和它们的客户（可以是其它组件）之间相互作用的方式，它可以使组件对象和客户程序不需要经过任何中介组件就能以一种统一的方式相互通信。而且，微软把 COM 技术用在了 Windows 系统的各个层次上。从操作系统级的 COM 对象一直到应用级的 COM 对象。

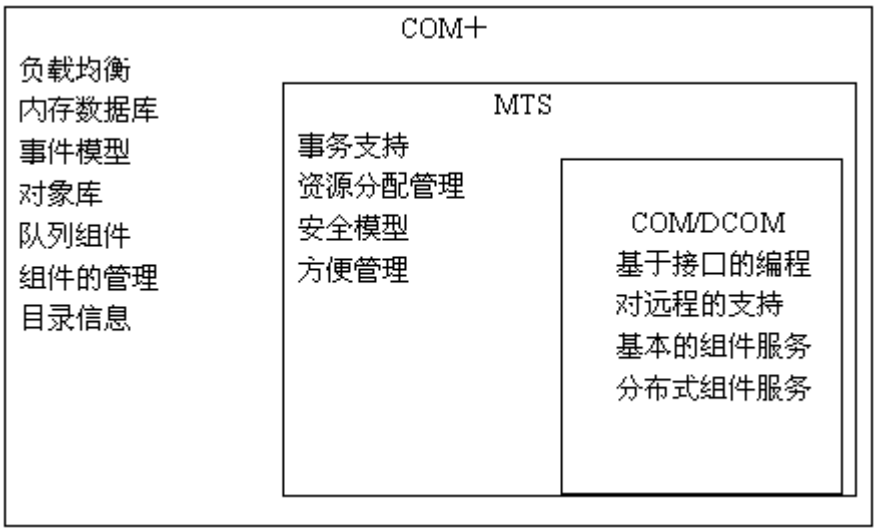
对于同一台机器上的 COM 对象，它具有很好的进程透明性，即 COM 对象和客户程序可以在同一个进程中，也可以在不同的进程中，但 COM 对象和客户程序不用考虑具体的通信细节，如系统关于进程间通信的一些规定，只要你遵守了 COM 标准，你就可以按照一般的函数调用方式来完成调用。但是对于不同机器、不同网络的 COM 对象，它们如何实现通信呢？我们需要手工来写一些 RPC（远程过程调用）之类的东西来完成吗？对此，微软发展了 COM，使 COM 的进程透明性拓展为位置透明性，形成分布式组件对象模型，成为 DCOM。DCOM 使用了 Object RPC 协议作为底层的通信协议。DCOM 是 COM 的无缝扩展，我们可以把它理解为 COM 和网络的结合。有了 DCOM，我们可以不用理会那些复杂的底层网络协议的细节，DCOM 为我们屏蔽了这些，我们仍然可以向以前一样，传递参数如同调用函数一样调用其它机器上的 DCOM 组件。DCOM 除了拥有 COM 的语言无关性、可重用性好等有点以外，还提供了位置无关性、安全性好等新优点。DCOM 可以作为分布式多层应用的基本架构，我们可以把业务逻辑层放在 DCOM 中实现，客户端只负责输入数据并把服务器的响应结果会传给客户端。但是 DCOM 由于牵涉到网络和安全性，配置起来比较复杂，令人头痛，这在部分程度上影响了它的使用。我们可以简单的说，DCOM 提供了一种访问网络上其它机器的 COM 对象的手段。

微软提供了一个什么样的环境来运行 COM、DCOM 组件呢？MTS（Microsoft Transaction Server）就是这样一个 COM/DCOM 对象的 Container 执行环境。MTS 是 Microsoft 在 Windows 平台的中介软件之一，它的主要功能是让 Windows 程序员能够开发以组件为导向的分布式应用系统，并且具备在同质和异质数据库之间对修改的数据进行两阶段提交的功能。由于 MTS 提供了事务管理、各种 Pooling 功能以及对象及时激活等功能，执行在其中的 COM/DCOM 组件将不用考虑这些事情，而只是专门执行企业逻辑代码。关于事务管理、安全机制、各种 Pooling 功能等这些底层的系统服务将由 MTS 自动提供，程序员不用再在 COM/DCOM 组件中加入这些代码。这些对于开发一个安全稳固的、具备扩展性的多层分布式应用系统至关重要。微软提供的这种 MTS 架构将会半强迫的使程序员开发以组件为导向的分布式应用系统。执行在 MTS 中的组件我们称之为 MTS 组件，MTS 组件事实上还是 COM/DCOM 组件，不过它与一般的 COM/DCOM 组件在建立方式和生命周期、效率等方面都有所区别。MTS 组件的另外重要的一点就是它有执行属性，执行属性这个概念就是把组件的一些执行行为独立出来，让组件的执行环境即 MTS 来管理。这样做的一个好处是一个组件可以在不同的应用程序中搭配不同的执行行为属性来完成特定应用系统的要求，这是微软的一个重要战略，从 MTS 开始，持续到 COM+，在将来的产品和版本中也将持续下去。声明性开发的主要目的是简化开发过程和提高产品质量。声明越多，开发者的应用程序就越简单，应用程序中开发者可能编入的错误就越少。

对照于我们前面提到的图，可以看到，在中间层，IIS 负责应用逻辑层 WEB 页面的管理，MTS 负责应用逻辑层 COM 组件的管理。MTS 在多线程支持下工作，实现对 COM 组件的分布式连接管理、线程池自动管理及高性能事务处理的监视。MTS 的最重要的目的之一就是帮助开发者节约资源。应用程序使用组件可以共享与数据库的连接，使数据库不再和每个活动客户保持一一连接，而是若干个客户通过共享组件和数据库连接，降低了数据库的负担，提高了系统性能。此外，客户通过组件访问数据库时，MTS 的安全管理机制可以按权限将特定组件授给不同的用户组，使商务活动的安全性和系统结构有机地结合在一起。

在 Windows 2000 之后，微软终于推出了 COM＋，它标志着微软的组件技术达到一个新的高度。在 Windows 2000 中，COM +中集成了 MTS。过去称为 MTS 的组件，现在称为已配置的 COM +组件（configured COM+ component）。COM＋的推出有机的统一了 COM/DCOM/MTS 的编程模型，形成一个功能强大的组件体系结构，并且把

DCOM/MTS 的各种优势以更为便捷的方式提供给用户。我们可以把 COM+认为是 COM/DCOM/MTS 的统一体。COM+继承了它们所有的优点，并且更容易开发、发布、维护。微软的开发人员对于什么是 COM+做个 3 个解释：一、COM+的基础是 COM；二、COM+在此基础之上增强了一些通用服务；三、努力使 COM+变成一种易于开发、易于发布、易于维护的模型。同时，COM+的推出也有对抗 SUN－IBM－Oracle 的 EJB 的意味。COM+倡导了一种完全基于构件开发（CBD）的理念。在 COM+的模型里，组件的地位得到空前的提高，它更强调应用层面上的组件，通过操作系统（COM+环境）的支持，使组件对象模型建立在应用层上，把所有组件的底层细节留给操作系统。COM+技术的一个核心理念就是操作体统本身提供完成许多面向企业开发者的通用基本模块（如线程、对象 Pooling、事务服务管理、事件服务管理等），让系统设计人员把精力集中在企业本身的业务逻辑上。如果读者想了解 COM+的这些具体机制如事务管理机制、安全机制等可以参看。从下面这张图中可以看出 COM+与 MTS、COM/DCOM 的关系：



可以看到 COM+不仅继承了 COM、DCOM、MTS 的许多特性，同时还新增了一些服务，如负载均衡、内存数据库、事件模型、对象库、队列组件、组件的管理、目录信息等。COM+新增的服务为 COM+应用提供了很强的功能，建立在 COM+基础上的应用程序可以直接利用这些服务而获得良好的企业应用特性。COM+是 Windows DNA 结构的核心。

三、从 Windows DNA 到.NET 的转变

微软总裁巴尔默曾说.NET 是微软对未来的一个最大的赌注，在赌互联网将无所不在，无线接入将无孔不入。顺便说一下，微软对未来的打赌好像总是能赢。在计算机还只是大型昂贵的设施时，盖茨就认为未来会有 PC 机，并且 PC 机将无处不在，进而发现了一个以前没有过的独立的"软件市场"。以及进而推出的 Windows 在赌人们不会满足于 PC 机的现有应用，人们会喜欢方便漂亮的东西。Windows NT 在打赌 PC 机将强大到足以支持企业级运算。有兴趣的读者可以看一下比尔.盖茨的《微软二十五年》。

但比尔.盖茨也有犯错的时候。在 95 年之前，当互联网兴起并如火似荼的时候，比尔.盖茨却认为网络应用并不是未来软件开发的方向。在当年的互联网四骑士（思科、AOL、SUN、网景）里面，人们看不到微软的影子。但网景的成功很快惊醒了微软，微软开始奋起直追，并凭借强大的实力很快从第一代互联网的追赶者变为第二代、第三代互联网的领先者。但这似乎并不足以满足微软的胃口，微软耗费重金打造的.NET 要成为下一代互联网的标准。

我们看一下遵从 Windows DNA 体系的 WEB 应用的缺陷。遵从 Windows DNA 体系的 COM/DCOM/COM+分布式应用可以将程序功能分布到整个网络上，DCOM 构造于 RPC 体系结构的最顶层，使用 DCOM 远比使用 RPC 容易的多，但是它仍然继承了 RPC 的一些缺陷。第一个缺陷就是：RPC 和 DCOM 都更适用于 Intranet 而不是 Internet。RPC 和 DCOM 要求的端口在防火墙内部，不太可能被打开。这种局限对于开发上线的 WEB 应用是一个很严重的问题。第二个缺陷是使用 COM/DCOM 需要注册或者发布，这会对应用程序产生很大的影响，所以它并不是一个理想的解决方案。这两个缺陷.NET 都可以利用 Internet 上的标准 XML、SOAP 来解决。第三个缺陷就是利用 ASP 开发 WEB 应用时，会将负责程序的脚本和 HTML 混杂在一起，导致页面的脚本语言结构十分复杂，逻辑不清晰，可读性差，

不仅给编程人员本身带来不便，也给系统的维护带来不小的困难，特别是当应用逻辑需求发生变动时，修改这些臃肿、晦涩的解释性脚本源代码真是味同嚼蜡。.NET 中的 ASP.NET 可以使代码和界面完全分离，并提供了基于组件的开发，是 WEB 应用的开发效率大为提高。第四个缺陷是 COM/DCOM 是平台相关的，只能基于 WINDOWS 平台。这让许多应用只能选择 J2EE 体系。微软的.NET 有望解决这个痼疾。有消息说，微软 2003 年将推出基于 LINUX 平台的.NET FRAMWORK。虽然有些人对此持怀疑态度，但理论上总是有可能的。

.NET 没有完全抛弃 WINDOWS DNA，实际上我们可以把它看做是从 WINDOWS DNA 演进而来的，它是 WINDOWS DNA 的继续和发展。WINDOWS DNA 是一个解决方案的平台，它关注的是如何使用微软服务器产品来解决业务问题。但是除了一个技术规范以外，WINDOWS DNA 并没有任何切实的东西。但是.NET 不仅有一套技术规范，它还包括了几个实实在在的产品，例如编译器、类库、甚至最终的用户程序。如 Windows .NET 是操作系统平台、.NET Framework 是运行环境、.NET 企业服务器为产品服务器、Visual Studio .NET 为编程平台。

在开发工具上面，由 Visual Studio 6 演进为 Visual Studio.NET，程序语言由 Visual Basic 6 演进为 Visual Basic.NET，成为面向对象的语言，添加了程序语言 C#，一个面向对象基于组件的语言，一个为.NET 平台量身打造的语言。另外对 C++也做了扩展。在操作系统及后台的服务器方面，WINDOWS 2000 演进为 WINDOWS.NET，提供下一代 Internet 稳固的执行环境。DNA Server 则演进为.NET Enterprise Server，其中包括 SQL Server、Biztalk Server 2000、Commerce Server 2000、Application Center 2000 等等。除此之外，.NET 还增加了许多新的特性，包括.NET Framework（一个通用语言执行环境，提供一套功能齐备的类函数库，以协助程序设计师处理和系统沟通的细节）、Internet 支持（提供了 ASP.NET、WEB Form、WEB Service 和应用服务结构，让 WEB 应用程序的开发与整合变得更加简单、易用）、WEB Service（提供了许多 Services，可以让开发者把这些 WEB Services 当成软件积木直接组合使用）、Orchestration（应用在整合跨应用系统、WEB Services 以及企业的交易流程）。

对于.NET 的其他细节，我们不再多说了。.NET 对于 DNA 有很大的改变，并且为了实现.NET 要做下一代互联网平台标准的目标，在无线接入、WEB 服务集成等方面作了很大的创新。从纵深的方向了解一下微软技术的演变，对于学习今天的.NET 会有帮助。如果有时间，我们还可以从横向技术（JAVA、CORBA、IBM 的策略等）的发展，看一下.NET 是如何产生并演进的（竞争会促使技术进步）。

了解过去是为了更好的服务于现在。？？？？？？ 罗曼.罗兰

张晓鹏－2000 年毕业于北京工商大学计算机专业，先后在安徽省蒙城县委宣传部、北京社会保障卡检测中心作为技术人员工作。现在北京航空航天大学软件开发环境国家重点实验室读研。愿意将学习和工作中的心得、体会和朋友们交流。

责任编辑：炒饭