Broadview®
www.broadview.com.cn

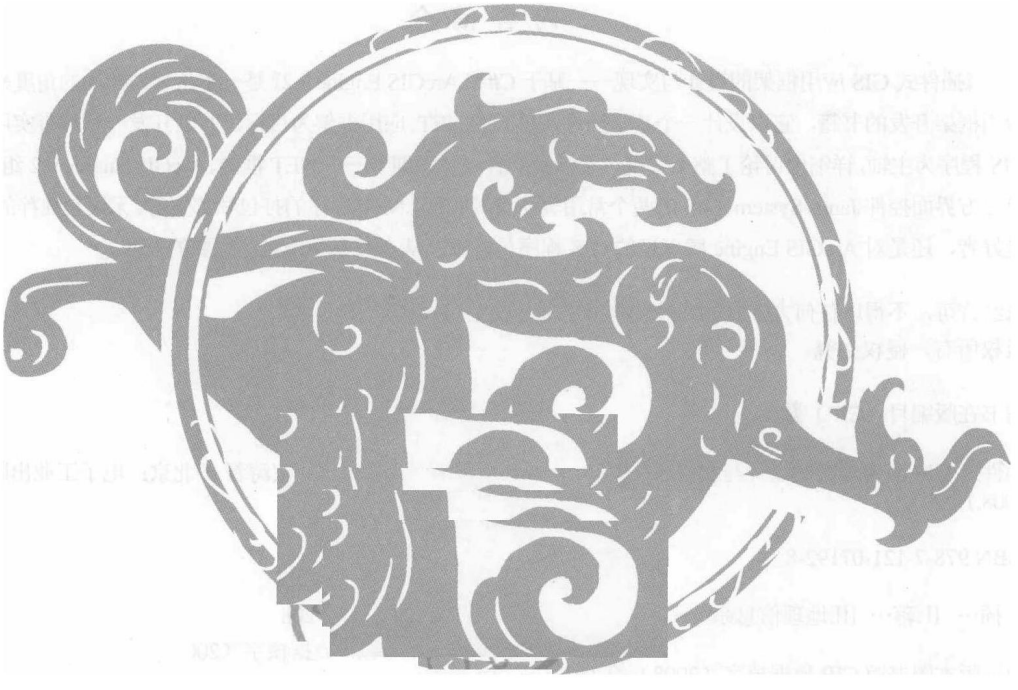
博文视点原创精品大系
GIS技术专家

# 插件式GIS应用框架的设计与实现

## ——基于C#和ArcGIS Engine 9.2

蒋波涛 著

電子工業出版社



# 插件式GIS应用框架的设计与实现

——基于C#和ArcGIS Engine 9.2

蒋波涛 著

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

## 内 容 简 介

《插件式 GIS 应用框架的设计与实现——基于 C#和 ArcGIS Engine 9.2》是一本从软件架构的角度来讨论应用框架开发的书籍，它以设计一个基于插件式机制的.NET 应用框架为目标，并以开发一款具有实用性的 GIS 程序为主轴，详细地讨论了整个实现过程涉及的各项知识细节——.NET 框架、ArcGIS Engine 9.2 组件集、第三方界面控件 Janus System、GIS 的两个常用算法及其实现、.NET 程序的打包和部署等。无论是纯粹的.NET 爱好者，还是对 ArcGIS Engine 感兴趣的 GIS 程序员，都能从本书中获得自己想要的东西。



图书在版编目（CIP）数据

插件式 GIS 应用框架的设计与实现——基于 C#和 ArcGIS Engine 9.2 / 蒋波涛著.—北京：电子工业出版社，2008.10

ISBN 978-7-121-07192-8

Ⅰ. 插… Ⅱ.蒋… Ⅲ.地理信息系统—应用软件，ArcGIS Engine 9.2 Ⅳ.P208

中国版本图书馆 CIP 数据核字（2008）第 115751 号

责任编辑：梁　晶

印　　刷：北京市通州大中印刷厂

装　　订：三河市鹏成印业有限公司

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

开　　本：787×1092　1/16　　印张：17.5　　字数：340 千字

印　　次：2008 年 10 月第 1 次印刷

印　　数：4 000 册　　定价：38.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

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。

服务热线：（010）88258888。

感谢所有在我工作、学习和生活中给予我帮助的人，

他们中的许多人促使我不断思考，

而本书，作为思考内容的一部分被整理出来，

奉献给大家。

# 推荐序

GIS（Geographic Information System，地理信息系统）早已不是什么新鲜的技术，特别是近些年来互联网已经深入我们日常生活的方方面面，人们有机会走进GIS的世界，才发现GIS原来这样精彩。在这背后，是GIS应用的日渐丰富，似乎世界上的一切都能与其扯上关系，其实这还真是一个再自然不过的道理，因为世间万物都有一个共同的属性，就是所处的位置，而地理位置，正是GIS的基础。

GIS的名头虽然不新鲜了，但GIS的前景仍然是很美妙的，毕竟从全世界的范围来看，GIS的应用朝气蓬勃，不断在崭新的领域创造价值，所以也吸引着广大的专家、工程师、学子将目光投向这一“超新星”。目前，国内几乎所有拥有地理专业的高校都设立了GIS这门课程，这预示着未来我们会遇到越来越多志同道合的伙伴在这一行当贡献各自的聪明才智。

平台软件是GIS应用的基本保证，而平台软件能够支持定制开发，这是对它的基本要求。在国内，在平台软件基础上做开发尤其受关注，对GIS的从业者来说，需要掌握的一个重要的技能就是GIS平台软件的二次开发。虽然“二次开发”看似比“一次开发”简单，但需要学习者掌握的技术也不少：除了GIS的基础知识外，还要懂得基本的程序设计方法和思路，了解至少一门编程语言的语法和开发环境，同时熟悉该GIS产品的开发原理和开发资源。

要求这么多的背景知识可难为了GIS的爱好者，好在本书的作者蒋波涛就是设身处地帮助大家思考如何破解这道难题的。本书不走常见的套路——要么侧重在组件式技术本身，要么空泛介绍GIS产品的主要组件——而是高屋建瓴地从设计模式和开发框架出发，剥茧抽丝，为读者展开一幅清晰透彻的GIS开发画卷。本书也包含了大量的基础知识，包

括组件技术、.NET 框架、C#语言、ArcGIS Engine 开发接口等，无论读者从哪个角度出发，我深信都能够从本书中咀嚼出滋味。

我要对希望快速汲取 GIS 知识力量的伙伴说，想象力比知识本身更重要，而想象力来自于对原理的掌握。想象不是凭空遐想，而是对原理的创造性延伸。本书的意义在于它并不只向读者介绍了一个 GIS 开发产品，况且 ArcGIS Engine 也仅仅是 ArcGIS 产品大家族中的一个小成员，更重要的是它为读者剖析了一种 GIS 二次开发的原理，为大家开启了一扇通向“想象力”的知识之窗。

王 昊

ESRI 中国（北京）有限公司技术总监

# 前 言

应用程序的开发是一项针对性很强的工作，程序的模块设计、层次划分、语言选择和系统部署都依赖最终用户的具体需求、软硬件环境甚至个人的喜好。因此，在应用程序开发过程中，项目人员必须花费大量的时间进行项目前期调研，编制不同阶段的需求分析和建设方案，然后才能进入编码阶段，根据具体需求开发各种功能组件。

如果每一个开发过程都是如此这般，每一次都要从头开始，舍弃以前的项目经验和成果，软件开发显然就是一件乏味且充满重复性的劳动。人类进化的显著特点之一是擅长学习已经存在的知识和经验，并将它们作为开发更广阔未知领域的工具。软件的开发也应该如此：吸取以往的开发经验和教训，尽量重复使用已经存在的组件和材料，以降低开发成本，缩短新技术的学习曲线并提高开发效率。

幸运的是，代码的重用从来不是一个过时的话题。在软件开发过程中，为了实现这一目标，函数库、类库、设计模式和框架等机制先后被提出，不断地为提高软件生产效率和代码质量而努力。而框架，作为这些重用机制中一种最高级的类型，在结构复杂、需求多变的各种规模程序设计中具有很强的吸引力——框架不仅仅提供了大量的可重用组件，而且提供了一种工作模式去协助开发人员扩展自身的功能。特别是插件式框架，它以一种更加灵活和宽松的方式实现了框架功能的无限扩展和功能聚合，更重要的是，这种扩展方式不会涉及框架本身的代码安全等核心问题。

本书的主题是设计一个基于插件式架构的 GIS 应用框架，GIS 开发是一种典型的“因势而变”工程，由于不同类型的用户在地理数据的存储、使用和部署方式上都有很大差别，根据实际需求定制开发是不可避免的。而插件式 GIS 框架，则是满足定制要求的最好方式之一——扩展方便、部署简单和高度可重用。

为了开发这个框架，本书使用了 C#2005 语言、.NET Framework 2.0 基础类库、ESRI 的 ArcGIS Engine 9.2 组件库和 Janus WinForm Controls 第三方组件库。

应用框架使用的开发语言是 C#，它总是在.NET Framework 中运行。因此，本书许多章节的内容将首先讨论为开发该框架某一部分而使用的关于 C#和.NET Framework 2.0 的相关知识。这些内容并非简单的语言语法，它还包括许多隐藏在语法后的使用技巧和实质内涵，即使其中的某些内容并没有直接用于框架设计，但它对于想要深入了解 C#和.NET Framework 高级主题的读者仍然具有很高的阅读价值。

插件式 GIS 应用框架使用了 ArcGIS Engine 组件库作为其 GIS 部分的二次开发包，因此本书中介绍了大量关于 ArcGIS Engine 的知识，尤其是 ArcGIS Engine 9.2 版本包含的新组件和新的扩展框架，这个重要的组件库使 GIS 开发工作变得更加容易和便捷，开发的程序功能也更加强大。但本书并没有系统地介绍其相关的内容。如果读者还不熟悉 ArcGIS Engine 或 ArcGIS 开发，可以参看拙著《ArcObjects 开发基础与技巧——基于 VisualBasic.Net》（武汉大学出版社，2006）或其他有关 ArcGIS 开发的书籍。

除了具体的技术，在本书中将要讨论的另一个重点是开发一个插件式 GIS 应用框架的模式，它包括设计模式和工作模式两部分。这些知识是开发者从初窥门径到登堂入室必须了解的内容，是隐藏在 GIS 应用框架中的精髓。

本书从一个插件式 GIS 应用框架的项目实例入手，循序渐进地介绍了开发这样一个框架的设计思考和代码实现过程，通过先理论后实践的方式，解答了为什么要这样做和怎样做的问题。首先设计插件式框架的核心——插件引擎，然后介绍基于插件引擎的框架宿主程序，接着开始设计框架的不同类型插件，这样一个插件式框架的雏形就建立起来了。为了让它具备 GIS 功能，本书继续深入介绍了宿主程序的高级设计内容——如何将 GIS 算法转换为代码和框架辅助类库设计的知识。最后，插件式 GIS 应用框架被打包和部署。

在项目介绍过程中，本书始终坚持了迭代开发的思想，没有在一开始就抛给读者一堆设计完整的代码。事实上，无论开发人员经验多么丰富，都免不了“再回首”的过程，因此，许多设计“缺陷”被刻意留在代码中，当读者深入学习后，会自然而然地发现这些缺陷，而此时读者理解这些缺陷和解决这些缺陷会变得更加胸有成竹。我认为这种组织方式更适合普通读者的学习曲线和理解习惯。

因此，这并不是一本关于 C#的语法书，它只挑选了 C#中的某些高级主题进行深入讨论；这也不是一本 ArcGIS Engine 9.2 开发包的参考手册，它介绍的是如何使用这些组件进行实际的程序开发；本书不是介绍如何使用拖曳控件方式开发应用程序，它更关注如何设

计和实现一个插件式框架；这也并不是一本只讲解如何使用 ArcGIS 组件的书籍，它还介绍了两个实用的 GIS 算法并基于 ArcGIS Engine 实现；最后，本书的软件成果并非一个只能供学习 ArcGIS Engine 而开发的简单 Demo，它完全可以投入实际应用。

本书适合以下读者阅读：

- 具有一定 C# 和.NET Framework 基础知识，并动手写过代码的读者；
- 已经熟悉 ArcGIS 开发技术，并希望开发水平更上一层楼的程序员；
- 对软件架构设计，特别是插件式框架设计有兴趣的读者。

最后，我希望这本书能够帮助读者有效地改进自己的思考方式，提高技术水平和开发能力。请读者在开发工具 VS2005 中认真地键入书中每一行代码，了解程序到底发生了什么变化。我相信，唯有如此，读者才能真正了解我的写作本意和迭代开发的精髓。

# 目 录

CHAPTER

第 1 章

# 走进插件式 GIS 应用框架

## 1.1 应用框架精讲

### 1.1.1 应用框架简介

“框架（Framework）是一个系统全部或部分的可复用设计，通常由一组抽象类和类之间的协作组成”[1]。

软件产品的开发是一项复杂的系统工程，随着它需要解决的问题复杂度的不断提高，软件产品的研发早已从过去“作坊式”开发演化到了当今符合一系列工业标准和规范的开发模式。软件产品面对的具体应用需求日益多元化、软件项目开发规模日益大型化，这两个因素促使软件开发团队的管理变得越来越复杂，软件开发项目的可控性变得越来越不稳定。

为了寻找“银弹”，IT 工业界采用了多种方法，它们包括制定各种软件开发标准和规范、发明具有更高生产力的编程语言、开发更好的编译器和运行时（Runtime）、提供功能更加强大的可分发组件库和探索更好的软件开发模式。但对于应用程序员而言，各种标准、规范和不断涌现的编程语言并不在自己可以控制的范围之内，大部分情况下，我们只能从软件工程的角度出发，在设计层面采用一些独特的软件架构和设计模式以达到我们期待的下列目的：

- 尽量提高软件的可重用性，避免不必要的重复编码工作。
- 增强组件的封装性。
- 提高软件的模块化程度。

1 Johnson R.E. Frameworks=(Component+Patterns). Communication of the ACM, 1997, 40(10): 39-42

- 不同功能模块之间能够无缝集成。
- 软件具有灵活的可扩展性。
- 软件产品的扩展和开发实现标准化。
- 软件产品具有面向不同应用层面的适应性和易移植性。

为了实现这些要求，在设计层面上，越来越多的软件产品开始采用应用框架（Application Framework）思想进行软件结构设计。应用框架已经是一个被广泛使用的术语，它成为软件开发中一种非常实用且常用的编程规范和设计架构。

我们肯定见到过许多自称“框架”的软件产品，也许有人会感觉不屑，有些代码量很少的程序居然也称自己是某种形式的应用框架？事实上，应用框架无关乎规模大小，就像房屋一样，摩天大楼和棚房都是房屋，只不过它们的规模和精巧度大不一样而已。

但无论如何，一个有资格称为“应用框架”的软件产品，必须体现应用框架的本质特征，即它能提供一个关联的、具有协调性的上下文环境（Context）和特定的工作模式（Work Model）供框架使用者访问和遵守。应用框架将为程序员提供开发应用组件的模板和一般方法，并隐藏了应用组件在框架后台的运行和协调过程。程序员将在拥有“说明书”和“烹饪原料”的情况下“烹制”自己需要的功能组件而无需担心“锅碗瓢盆”是否缺失和损坏。这样，开发人员就能更加专注于具体应用逻辑的代码实现而不用理会其他枝蔓问题，减少了工作量和旁枝末节的干扰。

## 1.1.2 应用框架的特点

应用框架相比其他程序结构而言具有五大主要特点：模块化、可重用性、可扩展性、简单性和可维护性[2]，尽管这些特点并非应用框架所完全特有，但它很好地保持了这些特点的平衡性，对于编写结构复杂、需求多变的大中型应用软件系统而言不失为最佳选择之一。

### 模块化

应用框架可以从逻辑上被划分为多个逻辑独立的层次或模块。模块化并非应用框架的独特之处，许多应用程序都具备这个的特点，它的好处是将整个应用独立为多个关联的模块，从而提高了应用的聚合性，降低了应用的耦合性。各个独立的模块通过统一的管道或协议进行通信互动，这样，当一个模块内部发生重大改变时，只要它的通信入口和出口保持不变，就不会影响到系统中其他模块的有效性和可靠性。

2 Xin Chen. 应用框架的设计与实现——.NET 平台. 温昱, 靳向阳 译. 北京: 电子工业出版社, 2005

就开发效率的层面而言，模块化是实现“人尽其能”的好方式。它将应用划分为不同的模块，各模块的开发可以由身负不同能力和技术的人员分别担当但同时完成。如 B/S 结构的应用，可以让美工负责界面端（UI 层）的设计，熟悉 ASP.NET、JSP 的程序员负责应用层，熟悉服务器编程的人员开发 Web Service，精通数据库设计的人员负责数据库。这样齐头并进的开发效率比程序员从头到脚大包大揽无疑高效得多。

### 可重用性

代码的可重用性是衡量代码质量的一个重要标志。无论是使用函数、类还是其他更高层次的重用模型，都是为了在某种程度上提高可重用性，使程序员避免编写重复的代码。而在应用框架中，可重用的不仅仅是函数和类，还包括多个类组成的一套逻辑和设计模式。可以简单地认为，设计框架的主要目的之一便是为了实现代码的重用。

可重用性并不是一件容易办到的事情，相反，设计具有良好可重用性的系统不仅困难而且需要不断进行迭代式设计。在面向对象语言中，一个类型被其他对象交互引用后，在修复该类型某个逻辑错误时很容易导致其他对象的逻辑错误呈指数级产生，这种所谓的“特征交互作用”是设计致命的逻辑错误之一，也是可重用性的最大敌人。

最简单的代码重用类型是使用函数库（Library），高质量的相关函数可以组合为一个可分发库，但这些函数之间并不一定存在紧密的联系，使用者必须从函数库中找出合适的函数，它的使用效率取决于使用者对函数库的熟悉程度和文档的完善程度。

比函数库复杂一些的是类库。类库提供了类（Class）而非单一的函数，一个类通常会包含相互有一些关联关系的函数，使这些函数更容易区分；除此以外，类库通常还会使用抽象类进行层次设计，这样能够使类的功能函数在使用上更加简单。

设计模式（Design Model）是另一种更高级的重用，它描述的是如何高效地创建面向对象系统的各个部分，如何在某种情况下使用特定的策略来解决具体问题，这种重用涉及面向对象的方法学。

框架重用是一种集合了类库、设计模式、工作模式（有时还有函数库）的重用模式，它重用的不仅仅是具体功能和解决问题的抽象策略，还包括解决问题的具体流程和规则。从重用程度而言，应用框架是最高级的重用模式。

应用框架与函数库的差别在于，后者只是提供了多组离散的、内部没有或有很少量逻辑关系的函数集，这些函数的调用顺序和调用关系并不明确，就好像使用 Win32 API 开发 Windows 程序一样，一个简单的窗体设计就必须有上百行代码，而且使用者还必须牢记

API 的使用顺序；而使用 MFC 框架编写窗体，代码数量就少得多，程序员需要考虑的内部细节也少得多，因为 MFC 已经将大量不必要暴露的细节隐藏在框架幕后，开发者不必关心那些通用的、繁琐的实现。除此以外，Visual C++还为开发者提供了多种工作模式，即开发不同类型程序的一般模板、向导和方法，所有这些都是应用框架与一般函数库之间的最大差别。

模块化与可重用性并没有直接的关系，模块化的软件并非都有很好的可重用性能，可重用的组件也并非一定被模块化。通常来说，模块化实现分治策略，可重用性是在不同的地方对功能的重复利用[3]。

### 可扩展性

可扩展性是应用框架最显著的特征之一，它意味着应用框架的功能具有生长的能力。没有扩展能力的应用框架毫无使用的价值和意义，因为框架本身就是为了提供一个统一的上下文环境给具体的应用使用。应用框架的可扩展性使我们能够基于一个平台实现不同的功能，满足不同的应用需要，当然，许多应用需要平台本身的支持。

框架的可扩展性主要是通过继承和聚合两种方式实现的。继承方式是指通过派生类继承基类，通过重用基类的功能并定义新功能的方式实现功能扩展；聚合方式是指调用不同的类型组合为一个新类型而扩展出全新的功能。这两种方法之间并不一定存在最好的模式，开发人员应该在合适的场合选择使用它们，但一般而言，我们推荐聚合而不是继承来扩展功能。

以 MFC 框架为例，扩展它的主要手段是在继承不同的基类上开发出具有新功能的类而达到扩展目的；在插件式框架中，插件类型则通过实现框架公布的接口完成扩展功能。

### 简单性

框架的简单性并不能用框架扩展组件开发的难易程度来衡量，事实上，没有哪一个框架能够保证基于自身开发的程序在逻辑设计和代码工作量上比不使用框架开发更有优势。框架的简单性体现在框架提供了一个明确的工作模式，即开发某种类型扩展插件的一般实现步骤，框架插件内在的关联和管理由框架本身控制而无需框架的使用者考虑，使用者可能根本不熟悉框架的内部协作结构，但扩展插件的开发可以很简单地在统一步骤的基础上进行升华和细化。

---

3 Christian Gross. .NET2.0 模式开发实战. 张凯锋，李彦娜，张广亮 译. 第一版. 北京：人民邮电出版社，2007