● 开发与编程实践系列丛书

AutoCAD ObjectARX
开发工具及应用

孙江宏
丁立伟　　编著
米　洁

清华大学出版社

北京科海培训中心

· 开发与编程实践系列丛书

AutoCAD ObjectARX 开发工具及应用

孙江宏　丁立伟　米洁　　编著

清华大学出版社

(京)新登字 158 号

内 容 提 要

本书以 AutoCAD R14 为基本开发环境，通过大量的编程实例，详细介绍了 Autodesk 公司最新推出的 ObjectARX 功能强大的开发工具。

全书共分 10 章，分别介绍了 AutoCAD ObjectARX 的基础应用，与 Visual C++的协同操作，AutoCAD 图形数据库的基本结构及操作，以及如何操作 AutoCAD 图形数据库中的符号表、词典、组和扩展记录的容器对象等，对原来开发环境难以实现的功能作了实例详解。

本书层次清晰，实例丰富，以其特有的结构及内容为 AutoCAD 开发人员提供了技术应用指南。

书　名：AutoCAD ObjectARX 开发工具及应用
编　著：孙江宏　丁立伟　米洁
出版者：清华大学出版社(北京清华大学校内，邮编 100084)
印刷者：北京市朝阳区科普印刷厂
发　行：新华书店总店北京科技发行所
开　本：16　印张：22.625　字数：489 千字
版　次：1999 年 2 月第 1 版　1999 年 2 月第 1 次印刷
印　数：00001～5000
书　号：ISBN 7-302-03396-X/TP・1840
定　价：30.00 元

前　言

众所周知，作为 CAD 工业旗帜产品的 AutoCAD，伴随着近年来整个 PC 基础工业的突飞猛进，正在迅速而深刻地影响着人们从事设计和绘图的基本方式。从始至今，AutoCAD 均是一种定位于全球各大领域和各类专业的通用微机 CAD 平台软件，其完备的系统开放性和丰富的个性化能力，是其能在各行各业应用中生机勃勃的基本要素。到目前为止，Autodesk 公司提供了 4 种二次开发手段：AutoLISP、ADS、VBA 及 ObjectARX，其中 ObjectARX 是全新的面向对象的开发手段，功能非常强大。

ObjectARX 编程环境

ObjectARX 是基于 AutoCAD R14 的一种新的编程环境，它提供了以 C++为基础的面向对象的开发环境及应用程序接口，能真正快速地访问 AutoCAD 图形数据库。通过在 CAD 应用系统开发中的应用表明，ObjectARX 的许多特性能够实现很多原来开发环境难以实现的功能，提高了开发效率，它是 AutoCAD 新一代的开发工具。

与以往的 AutoLISP、ADS 开发工具相比，ObjectARX 主要有以下几方面的特点：

- 开发环境
- ObjectARX 开发平台：

  AutoCAD R14.0
  Windows 95、98 及 Window NT

- 开发工具：

  Microsoft Visual C++ 4.2b 以上版本编译、调试集成化开发环境
  ObjectARX 开发工具包

面向对象的特性

针对 AutoCAD R14 推出的 ObjectARX 编程工具是采用全新的面向对象技术，其面向对象的数据体系及编程方法是其最吸引人的特点。

ObjectARX SDK 主要提供了以下几个类库：

- **AcRx 类库**　提供了系统级的类和 C++的宏指令集，用于 DLL 应用程序的初始化、连接及实时类的注册和一致性检查。
- **AcDb 类库**　提供可以直接访问 AutoCAD 数据库结构的类，包括实体等图形对象和层、线型等非图形对象。
- **AcGi 类库**　提供了用于显示 AutoCAD 实体的图形接口。

- **AcGe 类库** 该库主要被 AcDb 库的类使用，包括与通用线性代数和几何对象相关的类。
- **ADS 类库** C 语言的库函数，用于实体选择集操作、可编程对话框等操作，这些操作主要为了与 AutoCAD R12 兼容。

此外，ObjectARX 编程环境可利用 MFC 类库来编制丰富的 Windows 风格界面。并且，由于采用了面向对象的编程技术，开发人员可在其提供的基类上派生出自定义的类，给开发工作带来极大的方便。

ARX 是可编译的 C++程序，与 ADS 不同的是，ARX 以动态链接库(DLL)的形式与 AutoCAD 共享地址空间，直接调用 AutoCAD 的核心函数，并可直接访问 AutoCAD 数据库。除此之外，用户还可以在原有 ARX 系统上增加新的类，实时扩展原有类的函数。与 ADS 及 AutoLISP 相比，ARX 应用程序更稳定。

以上这些新特性，使得 ObjectARX 开发手段成为众多 Autodesk 二次开发商目前首选的开发工具，利用它开发出了丰富多样的应用软件，如 Machinical Desktop 等。

本书结构

本书主要介绍了 Autodesk 公司推出的 ObjectARX 开发手段，为广大 AutoCAD 开发人员提供较为详细的 ObjectARX 开发参考。

第 1 章主要对 Autodesk 提供的二次开发手段进行简单介绍。

第 2 章是以最简单的 ObjectARX 应用程序为例，说明在 Miscrosoft Visual C++ 4.2b 集成化开发环境下如何编辑、链接及编译 ObjectARX 应用程序，如何在 AutoCAD R14 中加载、调用 ARX 应用程序。

第 3 章介绍了 AutoCAD 图形数据库的基本结构，并对数据库中主要构成部分做了简单的介绍。

第 4 章和第 5 章详细介绍了 AutoCAD 图形数据库的结构、对图形数据库的操作等，并有大量的编程实例供读者参考。

第 6 章介绍了 AutoCAD 图形数据库对象——实体对象的公用属性和函数，并以编程实例讲解如何创建块、块插入和复杂实体以及选择和高亮显示子实体等数据库操作。

第 7 章介绍了如何操作 AutoCAD 图形数据库中的符号表 、词典、组和扩展记录等容器对象。

第 8 章介绍了 ObjectARX 提供的类库和开发时用到的一些重要的类，并详细说明了开发者如何从现有的对象派生出新的类，

第 9 章讲解了如何从 AcDbObject 类派生出开发者所需要的自定义类。

第 10 章讲解了如何从 AcDbEntity 类派生出开发者所需要的自定义实体类，并介绍了 AcGi 类库的用法。

读者范围

本书是我们编写的有关 ObjectARX 高级开发系列丛书之一，本书的特有结构及内容深

度决定了它的读者对象主要是面向 AutoCAD 的工程使用人员及基于 AutoCAD 平台进行软件二次开发的编程人员，写作目的就是要为他们在开发时提供很好的帮助参考，解决实际问题。

本书是集体创作的结果，是多人智慧的结晶。全书由陈士政老师组织编写，参加编写的人员还有张健、齐勋等。由于编写时间较为仓促，所以难免有错误和不足之处，敬请读者批评，以便日后改进。另外，热忱希望广大同仁能同我们进行交流，共同促进该项技术进步。

编者

1998 年 8 月

目 录

第 1 章　AutoCAD R14 开发系统概述

Autodesk 公司是目前世界上比较成功的 CAD 领域的开发商，其开发的 AutoCAD 一直是 CAD 市场中的主流产品。随着 AutoCAD 的日益普及，在其上进行二次开发的工具也相继由该公司推出。

本章简单介绍 Autodesk 公司针对 AutoCAD R14 提供的开发工具，以便读者全面了解和比较该公司最近提供的开发工具。本章后面部分着重介绍 AutoCAD R14 面向对象的开发系统(ObjectARX)、开发 ARX 应用程序的基本概念、方法及需求。

ADS 是从 AutoCAD R11 开始引入的 C 语言开发工具，ADS 不是 C 语言的一个子集，也不是基于 C 而编制的专用语言(如 Microstation 的 MDL)，但它包括了 C 的全部功能。C 是编译型语言，功能强大，因此 ADS 很轻易地克服了 AutoLISP 语言所固有的诸多缺点。ADS 程序不包含于 AutoCAD 内部，而是作为一组 AutoLISP 外部函数由 AutoLISP 解释器装入、解释并请求 AutoCAD 运行。

AutoCAD R13 和 AutoCAD R14 版都提供了面向对象的 Object ARX 开发系统，Object ARX 是用来开发 AutoCAD ARX(AutoCAD Runtime eXtension)实时运行扩展的一种新的编程环境，包括一系列的类库及头文件。编译后的 ARX 程序实际上是 Windows DLL 程序，与 AutoCAD 核心程序通信更为紧密，运行速度比 ADS 程序更快。

随着 AutoCAD R14 版的推出，Autodesk 公司推出了 Visual LISP 集成化编程环境，它不仅仅是 AutoLISP 的简单扩展，还在其中加入了面向对象的新特征，使利用 AutoLISP 进行 AutoCAD 二次开发的用户在编程时更为容易、快捷。

在 AutoCAD R14.01 版中，VBA 成为软件的标准安装组件，VBA 编程工具的引入为 AutoCAD 带来了第四种应用软件开发工具(前三种为 AutoLISP、ADS 和 ARX)，它可以使用 Active X Automation 组件。

图 1-1 反映了 AutoCAD 二次开发工具的演变。

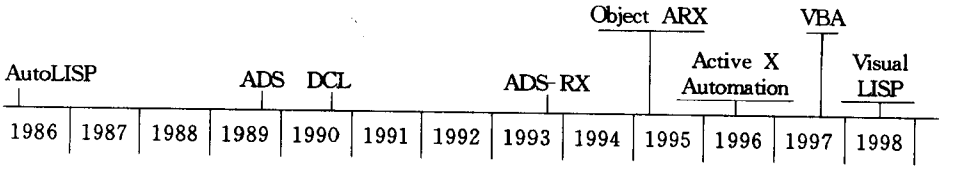

图 1-1　AutoCAD 二次开发工具的演变

1.1　Visual LISP 简介

Visual LISP 是继 AutoLISP、ADS、ObjectARX、Microsoft VBA 等 AutoCAD 开发工具之后 Autodesk 公司奉献给广大用户的又一强大开发工具。Visual LISP 是一个可视化的 LISP 语言开发环境，它是 AutoLISP 语言的扩展和延伸。基于 ObjectARX 技术的 Visual

LISP 提供了一组性能优越的面向对象的开发工具集，它使 AutoCAD 的 LISP 语言开发工具走进了新的时代。

Visual LISP 集成化开发界面如图 1-2 所示。

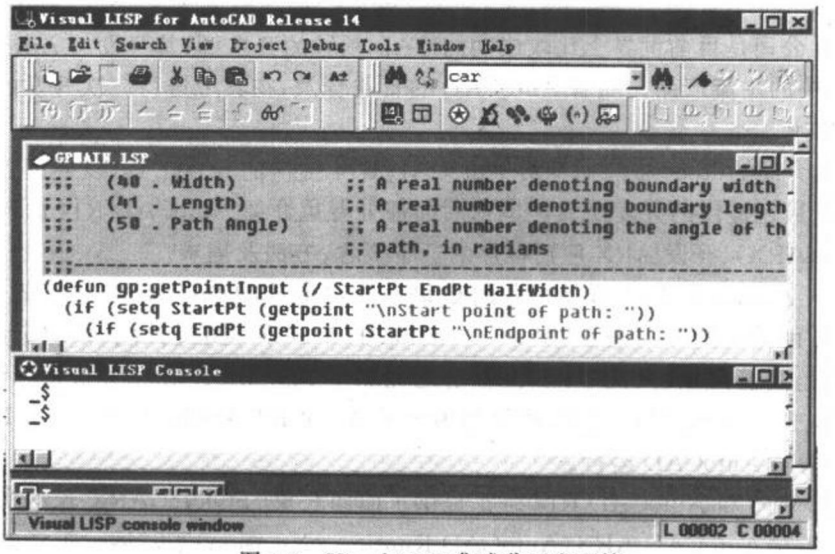

图 1-2 Visual LISP 集成化开发环境

Visual LISP 主体模块是一个 ObjectARX 应用程序，和其他 ARX 应用程序一样，该模块可以在 AutoCAD 环境中按需调入，具有即插即用的特征。Visual LISP 可以把 LISP 源代码编译成 ObjectARX 应用程序，从运行效率上看，编译后的程序比解释型的 AutoLISP 程序快 3～10 倍。Visual LISP 提供的与 AutoCAD ActiveX 对象的接口函数，以及操作对象事件反应器的函数等大大扩展了 AutoLISP 应用程序的功能。可以说，Visual LISP 应用程序的性能不仅仅是速度特性所能概括的。

1.2 ObjectARX 程序的特点

ARX 程序是 Windows DLL 程序，ARX 可与 AutoCAD 的另外两个开发工具 AutoLISP 及 ADS 共同工作，从效率和功能来说，ARX 应用程序远高于前两者。

1.2.1 程序结构以及与 AutoCAD 的通信机制

AutoLISP 完全包含在 AutoCAD 的内部，ADS 程序由 ads_init()初始化，它与 AutoLISP 的通信是一个被动地等待 AutoLISP 请求的无限循环，也就是说，ADS 程序与 AutoCAD 程序并不是直接通信，而是以 AutoLISP 为中介联系起来的。

ARX 程序本质上是地道的 Windows DLL 程序，而 AutoCAD 软件本身则是一个典型的 Windows 程序，ARX 程序与 AutoCAD、Windows 之间均采用 Windows 消息传递机制直接通信，ARX 程序通过调用 acrxEntryPoint()函数建立与 AutoCAD 消息传递的入口，在 acrxEntryPoint()函数中用 Switch 语句处理来自 AutoCAD 的各种消息。详见第 2 章诸节中的编

程实例及 ARX 应用程序与 AutoCAD 的通信机制介绍。

1.2.2　命令的注册与执行

AutoLISP 命令由 AutoLISP 解释器逐行解释并请求 AutoCAD 执行。

ADS 命令由 ads_defun()注册成为 AutoLISP 外部命令，作为 AutoLISP 外部函数由 AutoLISP 解释器装入、解释并请求 AutoCAD 执行，不能用 AutoLISP 命令函数或 ads_command()来激活。

ARX 程序是 DLL 程序，它与 AutoCAD 之间采用消息传递机制直接通信，并用 acrxEntryPoint()函数建立与 AutoCAD 消息传递的入口。AutoCAD 通过 acrxEntryPoint()函数来调用 ARX 程序，该程序也是由一组命令组成。ARX 命令通过 acedRegCmds()宏负责注册，一经注册，即被添加到 AutoCAD 的原始命令集中，与 AutoCAD 自身固有命令一样由 AutoCAD 本身执行。执行 ARX 应用程序所需系统开销最小。

1.2.3　面向对象的特性

在 AutoLISP 和 ADS 编程环境里，数据库直接反映为当前的 DWG 文件，用户通过 AutoCAD 命令来操作其中的实体，但不知道数据库的内部组织结构。从 ARX 开发者的角度来看，AutoCAD 图形数据库是一个对象管理器，它管理的是图形及其属性数据，开发者可以根据应用的需要添加和建立新的管理机制，并加入到这个体系中去。ObjectARX 应用程序以 C++为基本开发语言，具有面向对象编程方式的数据可封装性、可继承性及多态性等特点，用其开发的工程 CAD 软件具有模块性好、独立性强、连接简单、使用方便、内部功能高效实现以及代码可重用性强等优点，并且支持 MFC(Microsoft Fundation Class)基本类库，能简洁并高效地实现许多复杂功能。

1.2.4　ARX 环境的新特性

AutoCAD ARX 编程环境包括大量 C++库，允许用户开发 AutoCAD 应用程序，扩充 AutoCAD 类和协议，以及创建与内部 AutoCAD 命令一样操作的新命令。

ARX-RX 应用程序就是一种可共享 AutoCAD 地址空间和建立对 AutoCAD 直接函数调用的动态链接库(DLL)。由于精心设计的可扩展性，这些 ARX 库既提供了便于定义新类的宏，也能对实时库中现有的类添加其他功能。

下面列出了 ARX 环境一些主要的特点。

- **最佳性能**

采用 C++方法与 AutoCAD 对象交互，通过动态加载的 DLL 建立对 AutoCAD 直接函数调用。

- **实时可扩展性**

ARX 应用程序是动态链接库，它是 AutoCAD 进程空间的一部分，不仅运行快，而且 ARX API 使用也较简单。

- **重入机制**

用户定义的每条“内部”AutoCAD 命令均与 DLL 入口点无关。

· **事件驱动**

除了消息传送外，也能接收各种事件的通告。

· **其他**

ARX 应用程序也可通过 AutoLISP 和 ADS 来加载或卸载，使用虚拟的无 resbuf(结果缓冲区)结构替代用于构造应用信息的 Xdata 等。

1.3 AutoLISP、ADS 与 ARX 的调用机制

1.3.1 AutoLISP、ADS 与 ARX 应用程序的调用机制

AutoLISP 是一种解释型语言，只能简单地为 AutoCAD 添加新的命令。AutoLISP 对平台有一定的依赖性，但 AutoLISP 应用程序是一外部进程，与 AutoCAD 进行通信需通过 IPC(Interprocess Communication，进程间通信)。

AutoLISP、ADS 及 ARX 应用程序与 AutoCAD 之间的通信见图 1-3。

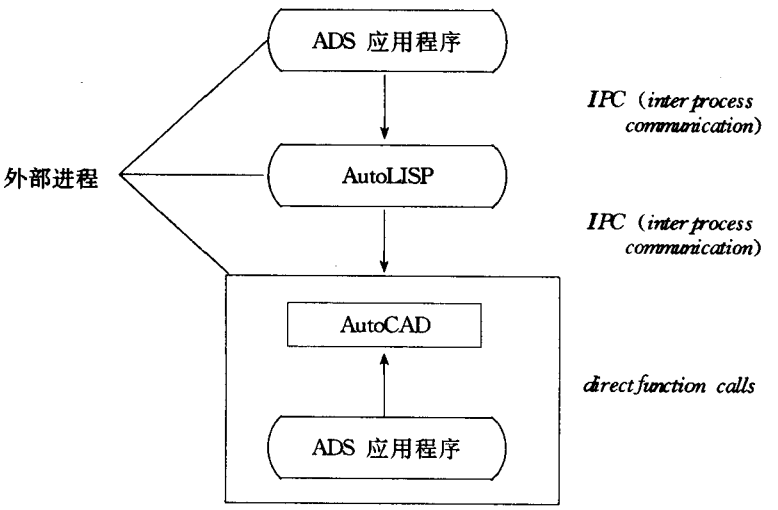

图 1-3 ARX、ADS 以及 AutoLISP 应用程序的区别

ADS 应用程序是用 C 语言编写且编译过的，它与 AutoLISP 应用程序一样，只不过是作为外部函数能通过 AutoLISP 解释器加载、执行，ADS 应用程序与 AutoLISP 之间的通信需利用 IPC 才能进行。

ARX 编程环境与 ADS、AutoLISP 的编程环境有很大的不同，最大区别是 ARX 应用程序是一种可共享 AutoCAD 地址空间和建立对 AutoCAD 直接函数调用的动态链接库(DLL)，因而 ARX 应用程序与 AutoCAD 之间的通信更加直接，运行效率更高。

开发者可在 ARX 编程环境中加入自定义的类，供其他应用程序调用。开发者所创建的 ARX 实体与 AutoCAD 内部实体有很大的不同，开发者可以加入新的协议来实时扩展 AutoCAD 的已定义的类。

ARX 部分类库中的函数是地道的 ADS 函数，这部分函数常被称为 ADS-RX 函数，与

标准的 ADS C 语言函数库完全一样，不同的是这些函数可共享 AutoCAD 的地址空间。这些函数与其他 ARX 类库中函数的使用没有任何区别。ADS 类库的主要作用如下：

- 构造实体选择集；
- 对实体选择集进行操作；
- 可编程对话框(DCL 对话框的控制)；
- AutoCAD 内部命令、函数的调用，如 ads_trans()函数、ads_command()函数及 ads_cmd()函数；
- 数据查询。

1.3.2　命令的注册

在 ARX 应用程序中，可以用 ads_defun()函数来注册新的命令，同时也可用 acedRegCmd()宏来注册。利用 ads_defun()函数注册的新命令首先要经 AutoLISP 解释器解释后才能执行，而利用 ARX 注册的命令可直接加入到 AutoCAD 内部命令集中。

命令的注册方式对命令的激活方式有影响。利用 ADS 库函数 ads_defun()注册的命令有以下特点：

- 可在 AutoLISP 中被调用或利用 ads_invoke()函数来调用；
- 不能在 AutoLISP 中利用 command 函数及 ADS 库函数 ads_command()来调用。

而利用 acedRegCmds()函数注册的函数则具有以下特点：

- 不能被 AutoLISP 和 ads_invoke()函数调用；
- 能在 AutoLISP 中利用 command 函数及 ADS 库函数 ads_command()来调用。

1.3.3　程序入口

ARX 应用程序与 ADS 应用程序同 AutoCAD 的通信机制是不同的。在 ADS 应用程序中是利用单一的无限循环等待 AutoLISP 的请求；而 ARX 应用程序中只有一个主要的入口用来处理消息，这样，当开发者注册一新的命令时，应用程序的入口将成为新的入口，当改写 ARX 类库中 C++类的虚函数时，这些函数就成为应用程序新的入口。

1.3.4　ADS 函数与 ARX 函数调用的比较

通常情况下，ARX 的 API 较 ADS 的 API 简单，在 ARX 应用程序中可以用如下代码来改变直线对象的层：

```
void
changeLayer(const AcDbObjectId& entId,
    const char * pNewLayerName)
{
    AcDbEntity *pEntity;
    acdbOpenObject(pEntity, entId, AcDb::kForWrite);
    pEntity->setLayer(pNewLayerName);
    pEntity->close();
```

```
}
```

注 在本章及以后章节中，程序段中的错误检查函数均忽略掉了，第 3 章 3.3.6 “容错处理”一节中详细介绍了如何正确使用错误检查函数。

在 ADS 中，实体的信息被保存在结构缓冲区链表中(“resbuf”)，改变直线实体的层需要以下四个基本步骤：

- 利用 ads_entget()函数来获得实体信息；
- 查找包含层信息的段；
- 改变段中的层信息；
- 调用 ads_entmod()函数来刷新结果缓冲区链表，从而使图形数据库改变。

相应的 ADS C 源程序代码如下：

```
void
changeLayerADS(ads_name entityName,
    const char * pNewLayerName)
{
    struct resbuf *pRb, *pTempRb;
    pRb = ads_entget(entityName);
    // 在此处不用检查 Rb 是否为空，因为所有的实体都有层特性
    for (pTempRb = pRb; pTempRb->restype != 8;
        pTempRb = pTempRb->rbnext)
    { ; }
    free(pTempRb->resval.rstring);
    pTempRb->resval.rstring
        = (char *) malloc(strlen(pNewLayerName) + 1);
    strcpy(pTempRb->resval.rstring, pNewLayerName);
    ads_entmod(pRb);
    ads_relrb(pRb);
    ads_retvoid();
}
```

完成相同事情的 AutoLISP 代码如下：

```
(defun asdk_changeLayerLISP(ename newLayer / eList)
    (setq eList (entget ename))
    ; 以新层名替换旧层名
    (setq eList
        (subst (cons 8 newLayer) (assoc 8 eList) eList))
    ;更新实体数据
    (entmod eList)
    (princ)
)
```

图 1-4 从程序运行速度、稳定性、效率、难易程度四个方面对 AutoLISP、ADS、ADSRX、ARX 编程 API 进行了比较。尽管 ARX 编程 API 从各方面来讲是最好的，但编程中的错误