第 9 章　自定义对象

用户可以定义自己的类来封装数据，需要通过 CRxDbObject 派生数据库对象；另外用于可以通过电子图版提供的点、线等通用的对象类型来派生自己的实体，如定义扳手类等。在派生的实体上除了具有自己形状特点外，还可以包含自己的一些数据，如扳手的材质等。

9.1　从 CrxDbObject 派生

9.1.1 说明

介绍自定义对象前，需要对电子图版中数据库对象的层次关系有所了解，这样才能理解后面的实际操作。电子图版中数据库对象的层次关系如下图 9.1 所示。

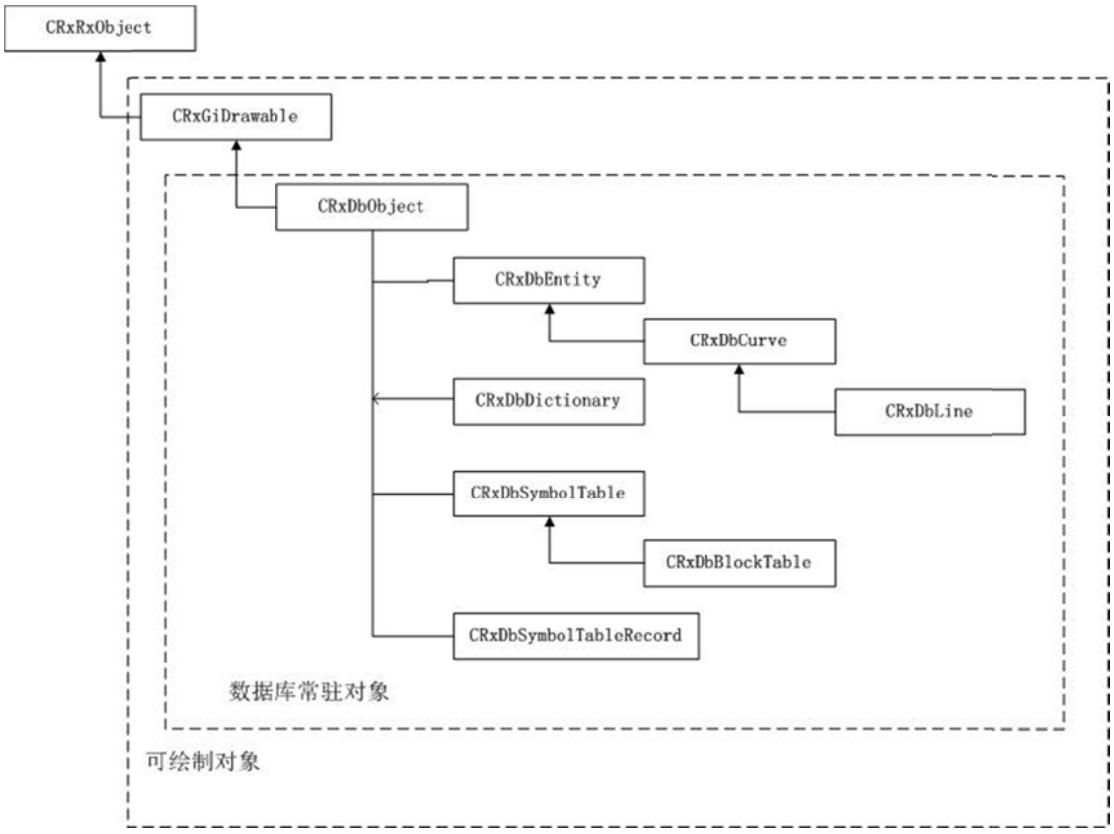

图 9.1 数据库对象的层次关系

从图中可知所有的数据库对象都是派生自 CRxDbObject，该类是所有数据库对象的基

类，它主要是实现对象运行时类型识别机制，提供一些用于类型识别的重要函数，他提供的函数主要有以下几个：

- desc（）：静态成员函数，返回指定类的类描述符对象
- cast（）：返回指定类型的对象
- isKindOf（）：用于判断对象是否属于指定类或者派生类
- isA（）；返回未知类对象的类描述符对象

CRxDbObject 派生的类都包含一个相应的类描述符对象，用 CRxRxClass 类表示，它包含运行时类型的识别信息，CRxDbObject 的派生类包含一个指 CRxRxClass 对象的指针（gpDesc),可以通过 CRxDbObject::desc()获取这个 CRxRxClass 对象指针，而 CRxRxClass 对象包含一个指向其父对象 CRxRxClass 的指针，这样就构成了类的运行时类层次表，如图 9.2 所示，可以通过 CRxDbObject::isKindOf()类判断对象是否是从某个类派生出来。

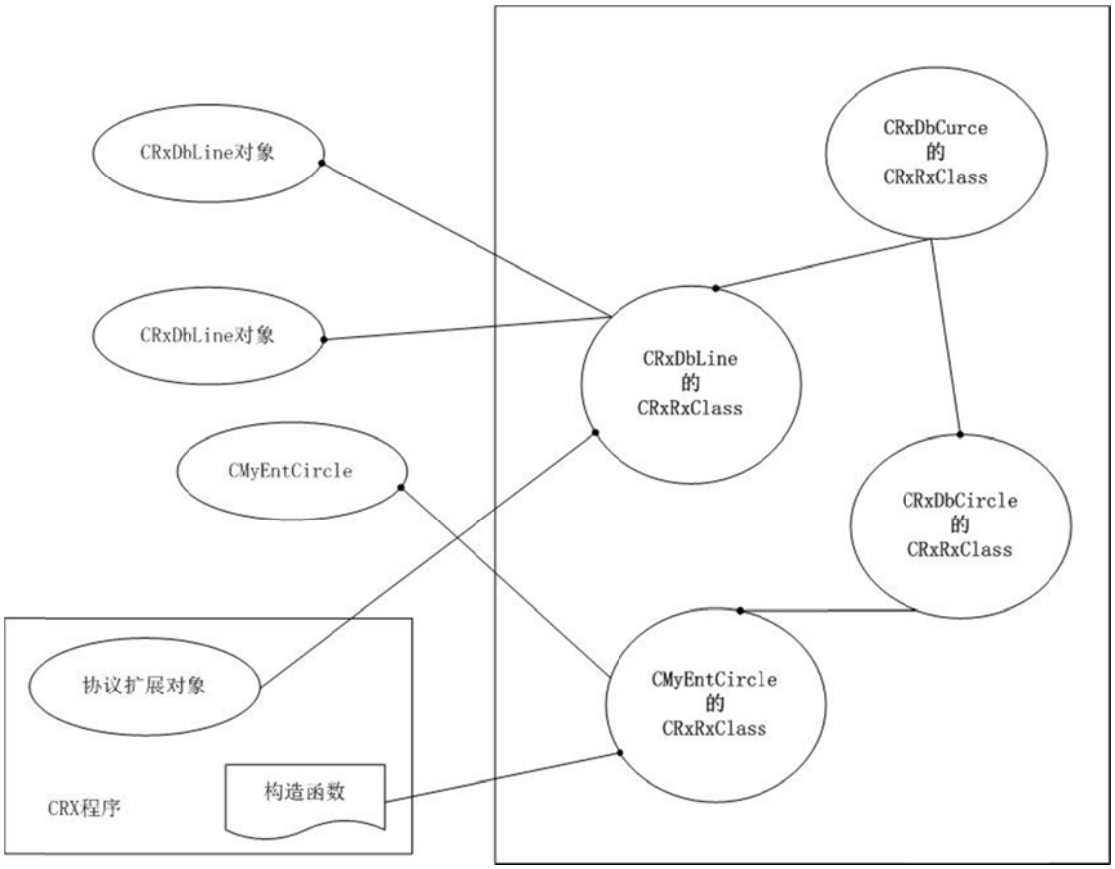

图 9.2 运行时类层次表

在派生类自定义中要实现运行时类的识别信息，也就是要重载上面提到的desc()、isKindOf()函数等，这个可以通过ObjectCRX提供的宏来实现，通过使用类声明宏CRX_DECLARE_DYNCREATE（CLASS_NAME）可以声明desc()，cast()，isA()函数。

```
Class  CMyClass:public CRxDbObject
{
  Public :
    CRX_DECLARE_DYNCREATE（CMyClass）;
```

```
}
```

该宏经过编译预处理后被扩展成以下代码：

```
public:
    static  CRxClass* gpDesc;
    static  CRxClass *desc();
    virtual CRxClass *isA() const;
    virtual CDraft::ErrorStatus getClassID(CLSID *o_pClsid) const;
    static void rxInit();
    static class_name* class_name::cast(const CRxObject *inPtr)
    {
        if (inPtr->isKindOf(desc()) == true)
        {
            return (class_name*)inPtr;
        }
        else
        {
            return NULL;
        }
    }
```

自定义类的静态函数 rxInit（）用于实现以下初始化操作。

1) 注册自定义类
2) 创建类的描述对象
3) 将类描述对象添加到类的描述字典中。

在crx程序中初始化函数中必须调用自定义类的静态函数成员rxInit（）来实现自定义类的初始化，然后调用全局函数CrxrxBuildClassHierarchy把该类添加到CRX运行类层次表中。另外在应用程序卸载时候需要调用deleteCrxRxClass（）把该类从CRX运行类层次表中删除，应用程序的初始化代码如下：

```
CxRx::AppRetCode crxrxEntryPoint (CxRx::AppMsgCode msg, void *pkt) {
        switch ( msg ) {
      case CxRx:: kInitAppMsg: return (On_ kInitAppMsg (pkt):
      //自定义类的初始化
        CMyClass::rxInit();
        //把该类添加到CRX运行类层次表中
        CrxrxBuildClassHierarchy();
        break;
      case CxRx:: kUnloadAppMsg: return (On_ kUnloadAppMsg (pkt):
         //该类从CRX运行类层次表中删除
        deleteCrxRxClass(CMyClass::desc());
         break;
        }
```

```
    return (CxRx::kRetOK) ;
}
```

除此之外，头文件中还需要使用一个派生类定义宏：

CRX_DECLARE_DERIVECONSTRUCT

相应的需要在类实现中添加两个对应的宏：

CRX_IMPLEMENT_DYNCREATE 和 CRX_IMPLEMENT_OBJDERIVECONSTRUCT(从Object派生对象使用)或 CRX_IMPLEMENT_ENTDERIVECONSTRUCT（从 entity 派生对象使用）

所有永久或者临时的图形对象都实现可绘制接口，封装该接口的对象可以通过绘制接口完成绘制，可显示的对象派生自CRxGiDrawable类，该类实现图形系统绘制协议。

CRxDbObject类执行文件操作协议，从该类派生的对象通过重载文件操作函数可以被保存未exb，或者从exb文件读入。

CRxDbEntity类是实体类，派生自CRxDbObject类，从该类派生的对象除了支持文件操作外可以通过重载绘制函数来按照用户的要求绘制图形。

9.1.2 思路

1.文件操作

CRxDbObject 派生子类要实现对文件的读写功能需要重载以下两个函数：

virtual CDraft::ErrorStatus saveData(CRxDbDatabase* i_pDataBase, IStream *i_pStm)
virtual CDraft::ErrorStatus loadData(CRxDbDatabase* i_pDataBase, IStream *i_pStm)

当电子图版调用保存命令时候，会调用数据库对象 saveData 函数； 如果调用 UNDO 命令取消操作时候，会调用数据库对象的 loadData。

在自定义类中重载文件操作函数时，必须首次调用 assertReadEnabled ()或者 assertWriteEnabled ()来检查对象处于正确的打开状态，然后调用自定义类父类的同名函数来提供对父类数据的重载。

9.1.3 步骤

（1）启动Visual Studio 2010，使用ObjectCRX向导创建一个新工程，其名称为AddAttribute。在项目中右键【Add/Class/c++ Class】，新建如下图所示的类CPipeAttribute

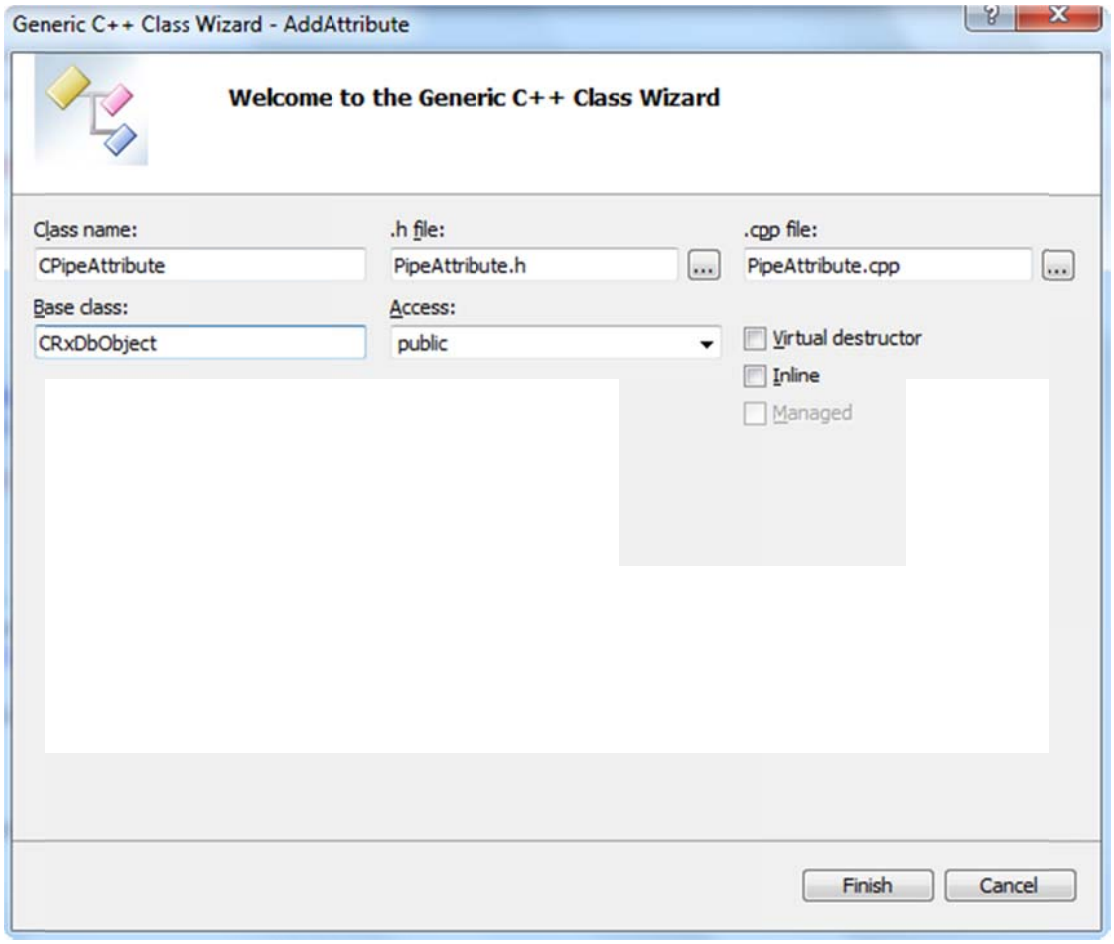

图9.3 创建自定义类

(2)在新建类的中添加四类成员变量：管径　m_dRadius;壁厚 m_dThickness;m_dDeep;材质 m_cMaterial[128];重载必要的函数，类成员声明如下：

```
class CPipeAttribute :
    public CRxDbObject
{
public:
    CRX_DECLARE_DYNCREATE(CPipeAttribute) ;
    CRX_DECLARE_DERIVECONSTRUCT(CPipeAttribute);

protected:
    static CAXA::UInt32 kCurrentVersionNumber ;

public:
    CPipeAttribute () ;
    virtual ~CPipeAttribute () ;
```

```
    //文件的读写
    virtual CDraft::ErrorStatus saveData(CRxDbDatabase* i_pDataBase, IStream *i_pStm);
    virtual CDraft::ErrorStatus loadData(CRxDbDatabase* i_pDataBase, IStream *i_pStm);

public:
    //管径
    double m_dRadius;
    //壁厚
    double m_dThickness;
    //埋深
    double m_dDeep;
    //材质
    TCHAR m_cMaterial[128];
};
```

（3）新建类重载函数的实现代码如下：

```
//-----------------------------------------------------------------------------
CAXA::UInt32 CPipeAttribute::kCurrentVersionNumber =1 ;

//-----------------------------------------------------------------------------
CRX_IMPLEMENT_DYNCREATE(CPipeAttribute, CRxDbObject)

CRX_IMPLEMENT_OBJDERIVECONSTRUCT(CPipeAttribute)


//-----------------------------------------------------------------------------
CPipeAttribute::CPipeAttribute () : CRxDbObject () {
    m_dRadius = 120.0;
    m_dThickness = 10.0;
    m_dDeep = 2.4;
    _tcscpy(m_cMaterial,_T("水泥管"));

}



CPipeAttribute::~CPipeAttribute () {
}
```

```
//-----------------------------------------------------------------------------

CDraft::ErrorStatus CPipeAttribute::saveData(CRxDbDatabase* i_pDataBase, IStream *i_pStm)
{
    //
    assertReadEnabled () ;
    //----- Save parent class information first.
    CDraft::ErrorStatus es =CRxDbObject::saveData(i_pDataBase, i_pStm) ;
    if ( es != CDraft::eOk )
        return (es) ;
    //----- Object version number needs to be saved first
    CAXA::UInt32 iVersion = CPipeAttribute::kCurrentVersionNumber;
    i_pStm->Write( &iVersion, sizeof(iVersion), NULL );
    i_pStm->Write( &m_dRadius, sizeof( m_dRadius ), NULL );
    i_pStm->Write( &m_dThickness, sizeof( m_dThickness ), NULL );
    i_pStm->Write( &m_dDeep, sizeof( m_dDeep ), NULL );
    i_pStm->Write( &m_cMaterial, sizeof( m_cMaterial ), NULL );

    return CDraft::eOk;
}

CDraft::ErrorStatus CPipeAttribute::loadData(CRxDbDatabase* i_pDataBase, IStream *i_pStm)
{
    assertWriteEnabled () ;

    CAXA::UInt32 iVersion = 1;

    i_pStm->Read( &iVersion, sizeof(iVersion), NULL );
    CPipeAttribute::kCurrentVersionNumber = iVersion;

    i_pStm->Read( &m_dRadius, sizeof( m_dRadius ), NULL );
    i_pStm->Read( &m_dThickness, sizeof( m_dThickness ), NULL );
    i_pStm->Read( &m_dDeep, sizeof( m_dDeep ), NULL );
    i_pStm->Read( &m_cMaterial, sizeof( m_cMaterial ), NULL );

    return CDraft::eOk;
}
```

（4）注册一个新命令 AddAttribute，使用上述定义的属性类。并将属性类的对象实例作为风格保存在命名字典中，代码如下：

```
static void CRXAddAttribute(void)
{
    CRxDbObjectId dictObjId,eId, attId;
    crx_name en;
    crx_point pt;
    if (  crxedEntSel(_T("\n选择管道（多义线）："), en, pt)!= RTNORM)
    {
        crxutPrintf(_T("\n选择失败，退出："));
        return ;
    }
    // 打开对象
    crxdbGetObjectId(eId, en);
    CRxDbEntity * pEnt;
    crxdbOpenObject(pEnt, eId,  CRxDb::kForWrite);
    if(!pEnt->isKindOf (CRxDbLine::desc ()))
    {
        crxutPrintf(_T("\n选择的不是管道（多义线），退出：" ));
        return ;
    }

    CRxDbDictionary *pNameObjDict, *pDict;
    crxdbHostApplicationServices()->workingDatabase()
        ->getNamedObjectsDictionary(pNameObjDict,
        CRxDb::kForWrite);
    // 检查所要添加的字典项是否已经存在
//  CRxDbObjectId dictObjId;
    if (pNameObjDict->getAt(_T("MyDict"), (CRxDbObject*&)pDict,
        CRxDb::kForWrite) == CDraft::eKeyNotFound)
    {
        pDict = new CRxDbDictionary;
        pNameObjDict->setAt(_T("MyDict"), pDict, dictObjId);
        pDict->close();
    }
    pNameObjDict->close();

    //  判断词典中的属性是否创建
    CPipeAttribute* pAttribute;
```

```
    crxdbOpenObject(pDict, dictObjId, CRxDb::kForWrite);
    pDict->getAt (_T("属性"),attId);
    if(attId!= CRxDbObjectId::kNull )//如果已经创建则输出数据
    {
        crxdbOpenObject(pAttribute, attId, CRxDb::kForRead);
        crxutPrintf(_T("\n管径：%4.2f " ),pAttribute->m_dRadius);
        crxutPrintf(_T("\n壁厚：%4.2f " ),pAttribute->m_dThickness );
        crxutPrintf(_T("\n埋深：%4.2f " ),pAttribute->m_dDeep );
        crxutPrintf(_T("\n材质：%s " ),pAttribute->m_cMaterial  );
    }
    else
    {
        //没有则创建属性
        pAttribute = new CPipeAttribute();
        pDict->setAt(_T("属性"), pAttribute, attId);
    }
    //关闭对象
    pDict->close();
    pAttribute->close();


} ;
```

9.1.4 效果

（1）编译链接程序，启动电子图板 2011，加载生成的 CRX 文件。选择一已存在的直线，为其添加在命名词典中的属性。

9.1.5 小结

学习本节的内容之后，读者需要掌握下面的知识点：

- 理解自定义对象的概念及其应用。
- 掌握电子图板数据库对象的层次关系。
- 掌握运行时的类层次。
- 掌握从 CRxDbObject 派生对象。

9.2 从 CrxDbEntity 派生

9.2.1 说明

CRxDbEntity 是从 CRxDbObject 派生的类，因此 CRxDbEntity 的派生类必须重载 CRxDbObject 类所有必须重载的函数，然后根据需要重载 CRxDbObject 类的其他虚函数和 CRxDbEntity 类的函数。

9.2.2 思路

1.派生自定义实体过程

CRxDbEntity 是所有具有图像表现的数据库对象的基类，它由 CRxDbObject 派生。自定义实体的步骤如下：

(1) 从 CRxDbEntity 派生一个自定义类。
(2) 重载必要的 CRxDbObject 类的函数。
(3) 重载要求的 CRxDbEntity 类的函数
(4) 重载支持自定义功能的其他函数。

2．worldDraw 函数

电子图板利用 worldDraw 函数来显示实体，对于任何有 CRxDbEntity 派生的类必须重载 worldDraw 函数，下面是函数的声明：

```
virtual  CAXA::Boolean    worldDraw   (CRxGiWorldDraw* mode);
```

当电子图板需要重新生成图形以显示实体时，使用如下方式调用函数：

```
if(!entity->worldDraw(pWb))
```

函数 worldDraw（）用于绘制实体的图形表达部分，与指定的模型空间和图纸空间的视口内容无关。

9.2.3 效果

本节不需要实例，参见 9.1

9.2.4 小结

学习本节的内容之后，读者需要掌握下面的知识点：

□掌握从 CRxDbEntity 派生对象。