国 家 开 放 大 学

学 士 学 位 论 文

题目：元器件管理系统

分部：陕西分部

学习中心：新城分校

专业：计算机科学与技术

入学时间：2018 年 3 月

学号：1861001211723

姓名：李谦

指导教师：蒋漪涟

论文完成日期:　2019　年　11　月

学位论文原创性声明

本人郑重声明：所呈交的学位论文，是本人在导师指导下，进行研究工作所取得的成果。除文中已经注明引用的内容外，本学位论文的研究成果不包含任何他人创作的、已公开发表或者没有公开发表的作品的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确方式标明。本学位论文原创性声明的法律责任由本人承担。

作者签名：　　日期：　　年　　月　　日

学位论文版权使用授权声明

本人完全了解国家开放大学关于收集、保存、使用学位论文的规定，同意如下各项内容：按照学校要求提交学位论文的印刷本和电子版本；学校有权保存学位论文的印刷本和电子版，并采用影印、缩印、扫描、数字化或其它手段保存论文；学校有权提供目录检索以及提供本学位论文全文或者部分的阅览服务，以及出版学位论文；学校有权按有关规定向国家有关部门或者机构送交论文的复印件和电子版；在不以赢利为目的的前提下，学校可以适当复制论文的部分或全部内容用于学术活动。

作者签名：　　日期：　　年　　月　　日

目　录

摘　要

当前，我国电子信息产业高速发展，各行各业电子信息化程度不断加速,各企事业单位都根据自己企业的情况，成立了信息化部门，专门针对本企业的情况研究信息化解决方案，采用现代的管理软件帮助企业较快发展。可以说企业的信息化系统的建设程度已成为企业现代化的标志。过去由于企业信息化发展落后，企业采用传统人工的方式管理库存中的电子元器件,这样的管理方式存在着许多缺点。

本文介绍一套对企业中的晶体管、电阻器、二极管等电子元器件实现信息化管理。 本文设计电子元器件管理系统具有良好的扩展性,可以帮助一些电子类企业进行元器件管理。技术层面本文采用Visual C++2015 做为开发工具，系统采用 C/S 结构，数据库采用 ACCESS2010 数据库，利用 ADO 控件与数据库进行连接。功能方面该管理系统可管理企业内各型号产品，可以建立各型号产品配件分系统，管理各类元器件使用情况。基于统一的数据中心，并可对元器件的产地、型号、批次等进行自定义规则查询统计，构建了一个完整、统一、协同的一体化管理平台，实现数据高度共享和准确，提高工作效率，帮助企业快速发展。

关键词：元器件；管理系统；面向对象；数据共享；ADO。

Abstract

At present, China's electronic information industry is developing rapidly, and the degree of electronic information in all walks of life is accelerating. All enterprises and institutions have set up an information department according to their own enterprises' conditions, which specializes in the research of information solutions for their own enterprises, and uses modern management software to help enterprises develop rapidly. It can be said that the construction degree of enterprise information system has become the symbol of enterprise modernization. In the past, due to the backward development of enterprise information, enterprises used the traditional manual way to manage the electronic components in the inventory, which has many shortcomings.

This paper introduces a set of information management for transistors, resistors, diodes and other electronic components in enterprises. The electronic components management system designed in this paper has good expansibility, which can help some electronic enterprises to manage components. At the technical level, this paper uses Visual C + + 2015 as the development tool, the system uses C / S structure, the database uses access2010 database, and uses ADO control to connect with

the database. In terms of function, the management system can manage all models of products in the enterprise, establish various models of product accessories sub-systems, and manage the use of various components. Based on the unified data center, it can query and count the origin, model and batch of components by user-defined rules, and build a complete, unified and collaborative integrated management platform to realize high data sharing and accuracy, improve work efficiency, and help enterprises develop rapidly.

Keywords:components; management system; object oriented; data sharing; ADO.

第一章 绪　论

1.1 课题背景

当前,我国电子信息产业高速发展，各行各业电子信息化程度不断加速,国家也提出中国制造 2025 计划，制造业是国民经济的主体，没有强大的制造业，就没有国家和民族的强盛。打造具有国际竞争力的制造业,是我国提升综合国力、保障国家安全、建设世界强国的必由之路，中国到 2025 年基本实现工业化，迈入制造强国行列。

目前,计算机已经在各行各业中都得到了广泛的应用，计算机化在企业中的应用会使得企业内部信息的获取、传达、管理更加便捷,使得企业的信息化水平得到了有效的提高。国内传统的制造企业型号产品多样，产品更新换代很快，每一个型号对应不同的配套产品,如何更好的管理和查询型号之间对各类元器件的使用情况，以方便满足企业的管理部门或物资采购部做好采购计划及质量跟踪。本人经过和企业的相关部门人员的沟通，深入调研了解他们的工作习惯和模式，结合大多数制造企业都是以型号作为单元进行管理的模式,对企业的元器件管理给出信息化建设方案。

1.2 课题研究

目前,国内很多制造企业对产品的一些物料和配件管理采用微软或金山软件的表格管理,这种单机文件管理的形式有很多弊病,表格管理容易造成数据孤岛,经常会出现乱码或数据丢失的情况,企业管理人员及其他应用部门查询和管理都不太方便，效率很低。

该元器件管理系统对企业中的 ARM 板、转换器、电源、半导体、晶体管、电阻器、二极管等电子元器件实现信息化管理，设计的电子元器件管理系统具有良好的扩展性，可以帮助企业进行元器件管理。通过元器件管理系统，规范元器件数据的录入，实现元器件数据的一次性录入，同时也可以基于已录入的数据进行分析元器件各种情况。

1.3 课题实现目标

针对每个企业的实际情况及管理模式，该管理系统可管理企业内各型号产品，可以建立各型号产品配件分系统，管理各类元器件使用情况。基于统一的数

据中心，对元器件的产地、型号、批次等进行自定义规则查询统计，管理部门可以基于软件系统做出决策，对常用和使用率较高的配件提前下单采购，对有质量问题的元器件进行查询跟踪，这样可更好的排除各型号产品是否有使用问题配件，为企业节约成本，降低企业库存负担。构建一个完整、统一、协同的一体化管理平台，实现数据高度共享和准确，提高工作效率，帮助企业快速发展。

第二章 开发环境概述

2.1 系统开发环境

系统开发环境选用微软的 Windows 7 操作系统、Microsoft Visual Studio 2015 开发环境和 Access2010 数据库。使用 VS2015 中的 VC++进行开发（MFC 框架）,它充分利用具有面向对象特性的 C++语言，开发出专业级的 Windows 应用程序，同时它提供了软件代码自动生成和可视化的资源编辑功能。

2.2 Visual C++2015 技术简介

Microsoft Visual Studio（简称 VS）是美国微软公司的开发工具包系列产品。Visual Studio 是目前最流行的 Windows 平台应用程序的集成开发环境。VS 是一个基本完整的开发工具集,它包括了整个软件生命周期中所需要的大部分工具，如 UML 工具、代码管控工具、集成开发环境(IDE)等等。所写的目标代码适用于微软支持的所有平台， Visual Studio 集成开发环境 (IDE) 将所有开发任务合并到一个工具中，Visual Studio 是高度可定制的，Visual Studio 将软件开发项目中涉及的所有任务合并到一个集成开发环境中，同时提供创新功能，使用户可以更高效地开发很多应用程序。多显示器支持，采用跨会话的连续布局以及数百项跨设备同步的可配置设置。

2.3 MFC 框架

系统用 VC++（MFC 框架）进行开发。MFC（Microsoft Foundation Classes，微软基础类库）是微软提供的类库（class libraries），以 C++类的形式封装的 Windows API，包含一个应用程序框架，以减少应用程序开发人员的工作量。其中类包含很多 Windows 句柄封装类和 Windows 内建控件以及组件的封装类。MFC 把 Windows SDK API 函数包装成几百个类，MFC 给 Windows 操作系统提供面向对象的接口，支持可重用性、自包含性以及其他 OPP 原则。MFC 通过编写类来封装窗口、对话框等其他对象，引入关键的虚函数（覆盖虚函数可改变派生类的功能）来完成。

2.4 系统数据库连接介绍

系统连接数据库的方式采用 ADO 连接方式。ADO（active data object，活动数据对象）原理上是一种基于 COM（组件对象模型）的自动化接口技术，并以 OLE DB（对象连接和镶入的数据库）为基层，经过 OLE DB 精心包装后的数据库访问技术，利用它可以编辑的创建数应用程序数据库。ADO 提供了一套非常简单，将一般通用的数据访问细节进行封装的对象。由于 ODBC 数据源也提供了一般的 OLE DB Privider，所以 ADO 不仅可以应用自身的 OLE DB Privider，而且还可以应用所有的 ODBC 驱动程序。

2.5 Microsoft Office Access 简介

Microsoft Office Access 是由微软公司开发的关系数据库管理系统。它结合了 MicrosoftJet Database Engine 和 图形用户界面两项特点，是 Microsoft Office 的系统程序之一，MS ACCESS 以它自己的格式将数据存储在基于 Access Jet 的数据库引擎里。它还可以直接导入或者链接存储在其他数据库或应用程序中的数据。

Access 有强大的数据处理、统计分析能力，利用 Access 的查询功能，可以方便地进行各类汇总、平均等统计。并可灵活设置统计的条件。比如在统计分析上万条记录、十几万条记录及以上的数据时速度快且操作方便，这一点是 Excel 无法与之相比的。

Access 用来开发软件，比如生产管理、销售管理、库存管理等各类企业管理软件，其最大的优点是：上手比较简单快捷，一般 OFFICE 都预装了 Access 数据库，不是计算机专业的人员，也能学会低成本地满足了那些从事企业管理工作的人员的管理需要。

2.6 开发模式（C/S 模式）

服务器-客户机，即 Client-Server(C/S)结构。C/S 结构通常采取两层结构。服务器负责数据的管理，客户机负责完成与用户的交互任务。在 C/S 结构中，应用程序分为两部分:服务器部分和客户机部分。服务器部分是多个用户共享的信

息与功能，执行后台服务，如控制共享数据库的操作等;客户机部分为用户所专有，负责执行前台功能，C/S 模式在出错提示、在线帮助等方面都有强大的功能，并且可以在子程序间自由切换。

C/S 结构在技术上已经很成熟，它的主要特点是交互性强、具有安全的存取模式、响应速度快、利于处理大量数据。

第三章 系统需求分析

3.1 系统的初步调查

元器件管理系统的基本功能需求主要有：

1） 实现元器件型号、分系统等基础数据录入；

2） 实现产品型号的新建，编辑

3） 实现型号负责人的权限管理

4） 实现型号的配套目录新建及编辑

5） 实现元器件数据的录入；

6） 实现元器件相关数据的修改和删除功能；

7） 实现元器件相关数据查询、统计等数据挖掘与分析功能；

8） 系统提供约束查询的新增、修改和删除。

9） 提供管理员修改信息，密码等功能；

10） 系统实现数据的便捷录入和输出功能。

元器件管理系统包括型号管理、用户管理、设备配套管理、元器件基本管理、元器件通用查询以及元器件统计汇总等多个方面。

3.2 可行性分析

3.2.1 要求

功能：为规范企业元器件数据的录入，实现元器件数据的一次性准确录入，同时也可以基于已录入的数据进行分析元器件各种情况，因此需要建立一套合理、有效、规范的管理系统来进行管理；

普通用户端：包括设备配套管理、元器件基本管理、通用查询、元器件比例汇总、元器件统计、国产化比例、质量等级统计等功能；

管理员端：包括型号管理、用户管理、配置管理等功能。

输入：用户在系统中能快捷方便的输入或者选择元器件相关数据。

输出：能按照功能要求在页面上显示所需要的内容并生成相关图表。

安全与保密：登录时需要进行用户名、密码验证；对于不同权限的用户设置不同的权限。

3.2.2 系统可行性分析

元器件管理系统的可行性分析包括以下两个方面的内容。

1.经济上的可行性

该元器件系统开发需求相对简单，加上现在企业都具备成熟的软硬件环境，企业中人员也都有一定的计算机应用基础，在软硬件的支出上很有限。该系统的开发也并不是十分的复杂，该系统的开发周期较短，企业的相关应用部门人员，只需要通过简单的软件培训就可上手，不会对企业的正常生产和运营造成影响，在经济上是可行的。

2.技术上的可行性

在 Visual Studio 开发环境下，利用 VC++（MFC） 框架下搭建元器件管理系统，采用具有面向对象特性的 C++语言进行编码开发；MFC 框架具有大量的大量 Windows 句柄封装类和 Windows 内建控件和组件的封装类，能够为开发者提供快速开发程序，提高了软件的开发效率。

数据库采用 Access2010 数据库，因其存储方式简单，操作界面友好；Access 基于 Windows 操作系统下的集成开发环境，该环境集成了各种向导和生成器工具，极大地提高了开发人员的工作效率。

3.2.3 系统主要功能设计

元器件管理系统主要功能设计包括：

1、管理员：型号管理、用户管理、配置管理等。

2、普通用户：包括设备配套管理、元器件基本管理、通用查询、元器件比例汇总、元器件统计、国产化比例、质量等级统计等。

3.3 系统结构化分析

3.3.1 数据字典

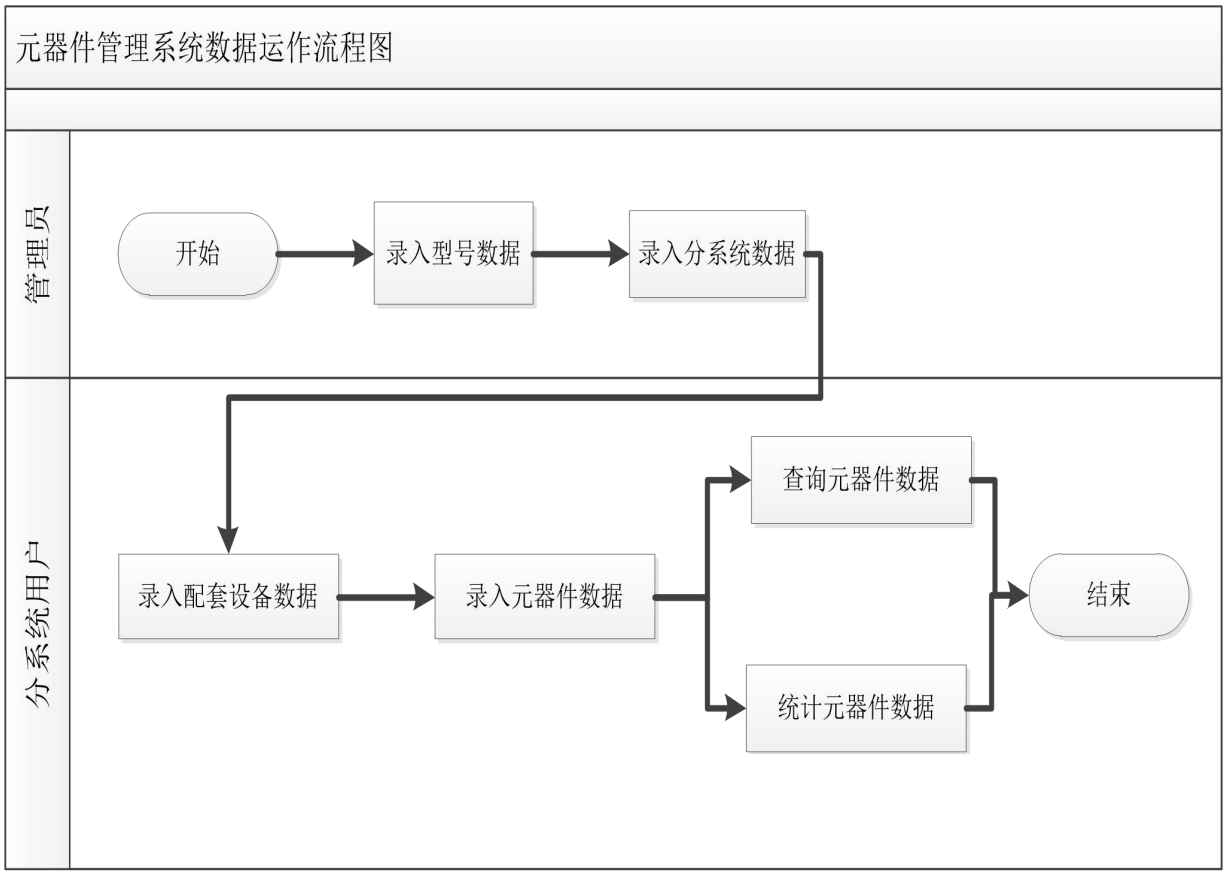
根据需求分析将数据库设计为 5 张表。业务数据字典描述如下：

型号信息	包含型号标识、型号名称、创建时间等
分系统信息	包含分系统标识、型号名称、分系统名称、创建时间等
用户信息	包含用户账号、用户名称、用户密码、创建时间等

配套设备信息	包含设备标识、型号名称、设备名称、设备代号、研制单位、分系统类型等
元器件信息	包含元器件标识、型号名称、设备代号、产品编号、元器件名称、元器件规格、规范号、封装类型、质量等级、质量等级分组、计量单位、数量、生产厂家、使用单位、国产 OR 国外、是否首次选用等
其他	

3.3.2 数据流程图

元器件管理系统使用对象是管理员和用户，数据运作流程图如下所示：



3.4 系统功能描述

元器件管理系统包括型号管理、用户管理、设备配套管理、元器件管理、查询统计管理。通过构建元器件管理系统，规范元器件数据的录入，实现元器件数

据的一次性录入，提高工作效率。

型号管理：型号数据的增、删、改操作。

用户管理：分系统数据的增、删、改操作。

配套设备管理：配套设备的增、删、改操作。

元器件基本管理：元器件数据的增、删、改操作。

配置管理：元器件录入过程中的约束选取内容配置。

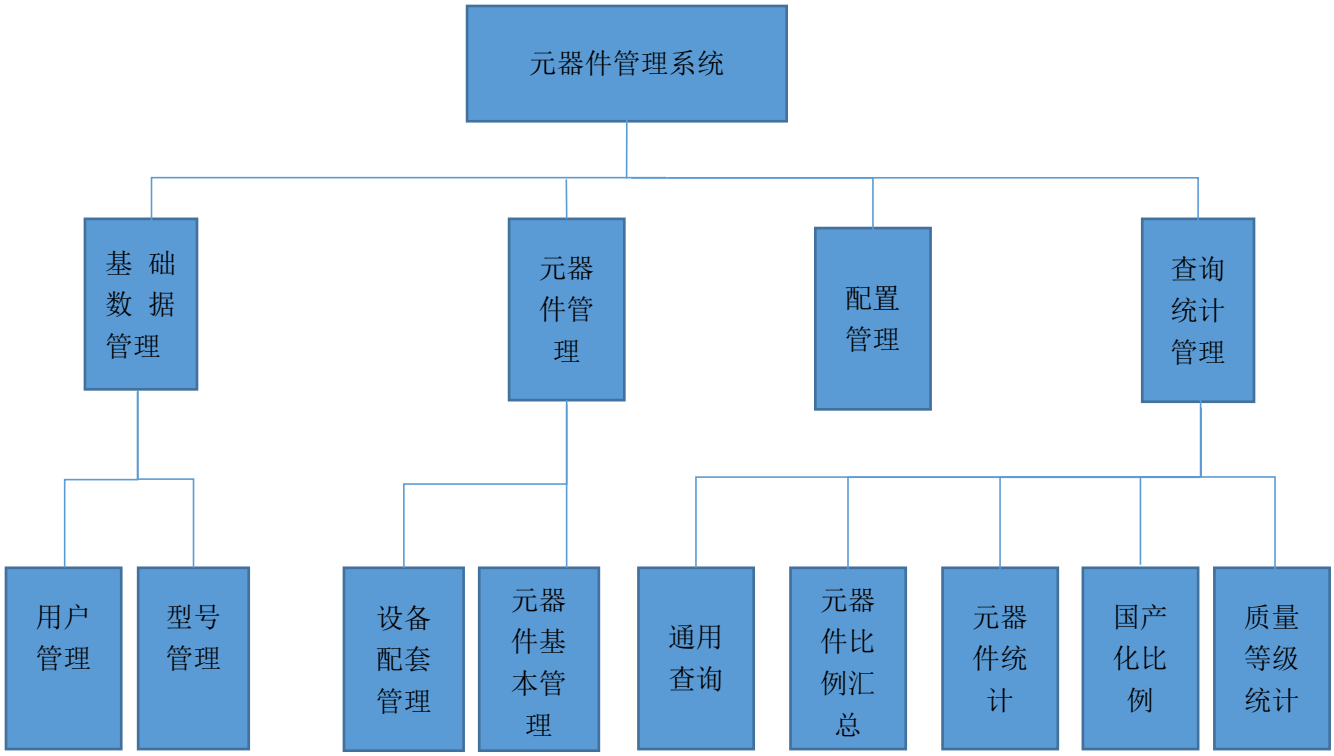
通用查询：根据用户输入各种条件，查询满足要求的结果记录。

统计汇总：元器件统计、国产化比例、质量等级、元器件比例汇总等统计汇总功能。

第四章 系统总体设计

4.1 元器件管理系统设计

元器件管理系统是基于 C/S 结构化的元器件管理系统。是用来管理元器件数据的软件，内部包括型号管理、用户管理、设备配套管理、元器件管理、查询统计管理。系统功能图如下所示：

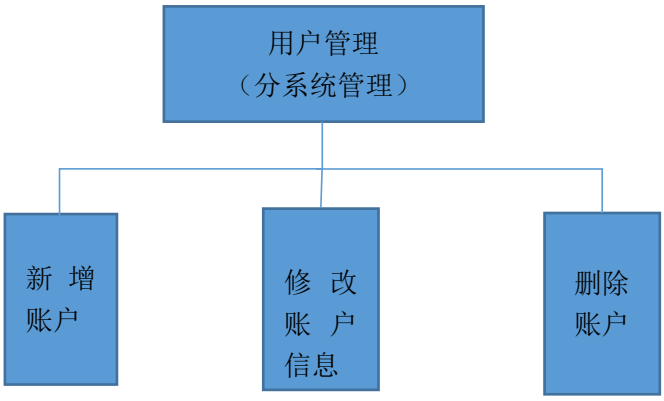


4.2 基础数据管理

基础数据管理包括型号管理和用户管理。基础数据管理模块为元器件管理系统提供型号、用户数据基本数据。

4.2.1 用户管理

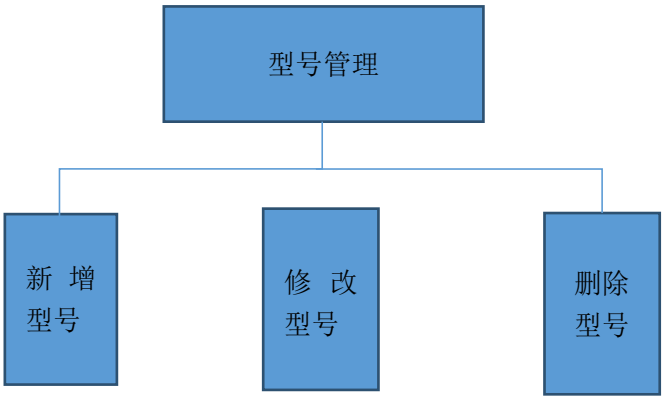
用户管理是采用总体和分系统账号进行管理，总体是某个型号的总负责账户，分系统是负责某个型号下对应模块的元器件信息维护的账户。用户管理也称为分系统管理。例如某某型号下分系统名称包含总体电路，测控通信，仪表照明，数管，结构机构，热控，二极管等。用户管理包含分系统账户的新增、修改、删除功能。针对分系统账户已录入元器件数据将无法进行删除。



新增账户是针对某系统进行账户的新增，输入用户名以及密码信息。修改账户信息是对已新增的账户信息进行修改。删除账户是对已存在的账户进行删除。

4.2.2 型号管理

型号数据是最基础的数据，是针对即将准备研制的实物进行命名，是元器件系统的数据源头。分系统是针对某一个型号进行建立的。型号管理包含型号的新增、修改以及删除操作。

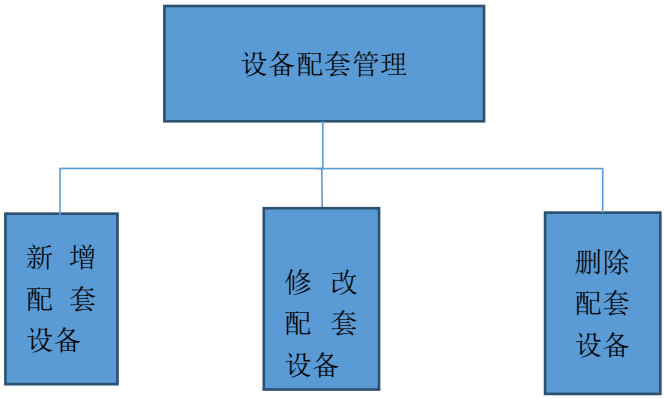


新增型号进行新增型号名称。修改型号信息是对已新增的型号名称进行修改。删除型号是对已存在的型号进行删除，型号下存在分系统将无法进行删除。

4.3 元器件管理系统功能

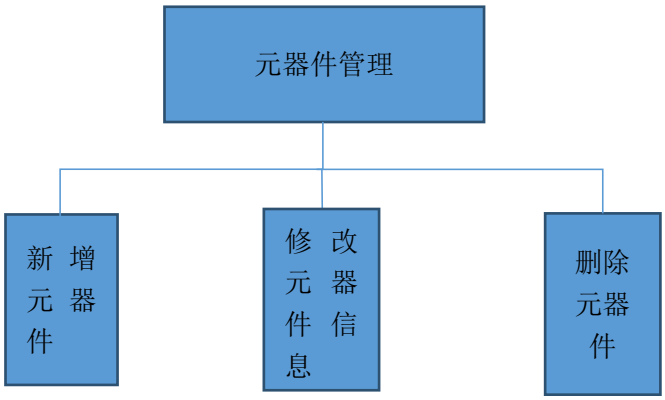
4.3.1 设备配套管理

设备配套是针对某型号下的分系统而言的，也可认为是归类在分系统下的设备。设备配套管理包含配套设备的新增、修改、删除功能。针对设备下已录入元器件数据将无法进行删除。



4.3.2 元器件基本管理

元器件基本管理是元器件系统最主要的数据，是归类在配套设备下的元器件数据。元器件基本管理包含元器件的新增、修改、删除功能。新增元器件可以新增元器件信息，如元器件分类、元器件名称、型号规格、标准规范号、质量等级、是否国产、计量单位、封装形式等。修改元器件信息是对已新增的元器件各项属性进行修改。删除元器件是对已存在的元器件进行删除。在录入元器件各项属性时可通过下拉框方式进行选择，也可直接录入信息。



4.4 配置管理

元器件录入过程中，下拉框的数据通过预先定义的数据进行配置的。

格式如下：

[约束属性]

数量=X

属性 X=XXX

[XXX]

数量=X

约束 X=XXX

配置示例如下所示：

[约束属性]

数量=2

属性 1=元器件分类

属性 2=质量等级

[元器件分类]

数量=11

约束 1=二极管

约束 2=晶体管

约束 3=光电器件

约束 4=单片电路

约束 5=混合电路

约束 6=表面波器件

约束 7=其他微电路

约束 8=继电器

约束 9=电连接器

约束 10=晶体器件

约束 11=电阻器

[质量等级]

数量=5

约束 1=CAST C

约束 2=CAST B

约束 3=CAST A

约束 4=CAST S

约束 5=SAST

.....

4.5 ADO 数据库连接

系统连接数据库的方式采用 ADO 连接方式。具体连接步骤如下：

1）导入 msado15.dll,MSJRO.DLL，一般位于 C:\Program Files\Common Files\System\ado\目录下，重命名 EOF 和 BOF 防止 ADO 库与项目发生冲突。

```
#import "C:\Program Files\Common Files\System\ado\msado15.dll" rename("EOF", "EndOfFile")

#import "C:\Program Files\Common Files\System\ado\MSJRO.DLL" no_namespace rename("ReplicaTypeEnum", "_ReplicaTypeEnum")
```

2）初始化 COM 组件。

```
CoInitialize(NULL);
.....
```

```
CoUninitialize();
```

3）创建一个连接对象，再调用其 Open 方法打开数据库。

```
_ConnectionPtr ConnAccess = NULL;
HRESULT hr = ConnAccess.CreateInstance(_uuidof(Connection));
sprintf(chFileName,"软件运行路径");
strcat(chFileName,"\\此处为 Access2010 数据库文件名.mdb");
sprintf(chConnStr,"Provider=Microsoft.Jet.OLEDB.4.0;Jet
OLEDB:Database Password=%s;Data Source=%s","xxx",chFileName);
HRESULT hr = ConnAccess->Open(chConnStr,L"",L"",-1);
```

4）使用连接对象的 Execute 方法执行 SQL 语句，使用记录集对象存储执行结果信息。

```
try
    {
        if (pOutArr)// 此处为接口函数输入参数，空类型指针，存储执行结果或为 NULL
        {
            rstmp =
ConnAccess->Execute((_bstr_t)strSQL,NULL,0);
            *(_RecordsetPtr*)pOutArr = rstmp;
        }
        else
        {
            ConnAccess->Execute((_bstr_t)strSQL,NULL,0);
        }
    }
    catch (...)
    {
        CString StrTemp = "";
        StrTemp.Format("错误 SQL 语句：%s",strSQL);
    }
```

5）读取记录集对象内容。

```
while (!pRecord->adoEOF){
    varfield = pRecord->Fields->GetItem(L"SUSERNAME")->Value;
// varfield 为_variant_t 类型
    if (varfield.vt != VT_NULL)
    {
        tmpStr    = varfield.bstrVal;// 例如 CString 类型 tmpStr
```

```
            // ...
        }
        pRecord->MoveNext();
    // 下一条记录
    }
```

4.6 查询统计管理

4.6.1 通用查询

支持按照条件查询元器件清单，可在查询界面输入各项查询条件后即可输出查询结果。抗总剂量、抗单粒子 SEL、SEU 可进行数字判断。

4.6.2 元器件比例汇总

统计元器件国产/进口比例图,用来分析国产化的元器件占比。

4.6.3 元器件统计

统计元器件各种类型的数量，以便需要备用多少元器件。

4.6.4 国产化比例

统计不同种类元器件国产化比例，用来分析国产化的元器件占比。

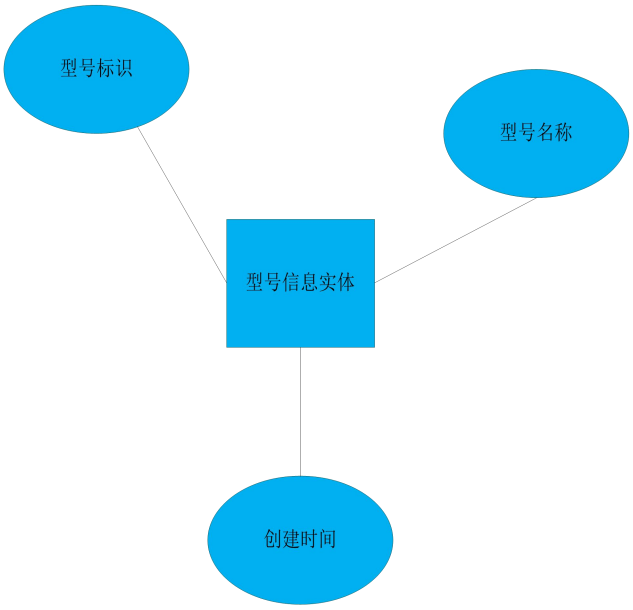
4.6.5 质量等级统计

统计元器件质量等级，分析使用的元器件采用了哪些标准。

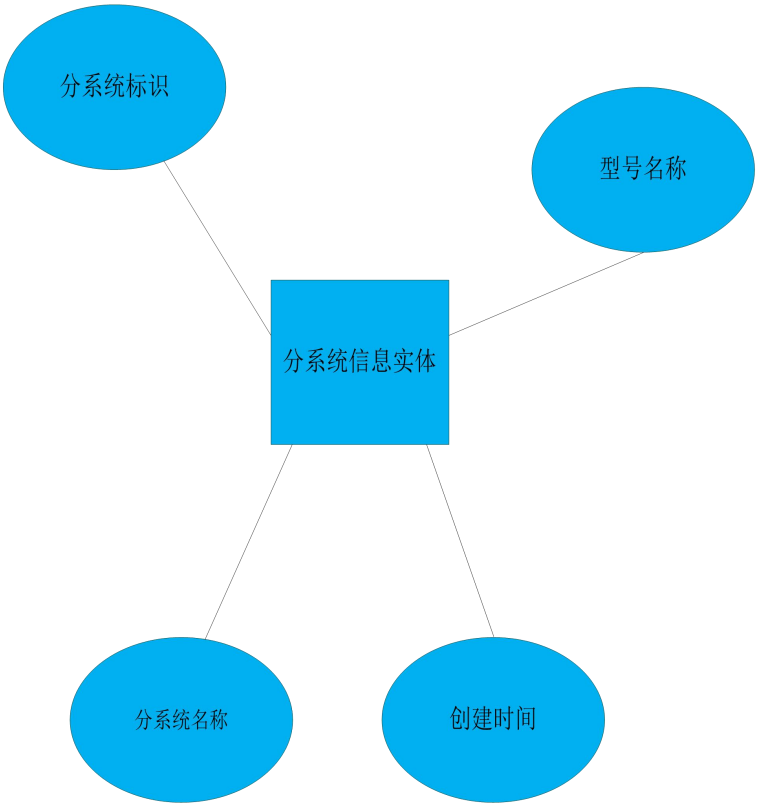
4.7 系统数据结构

4.7.1 逻辑结构设计

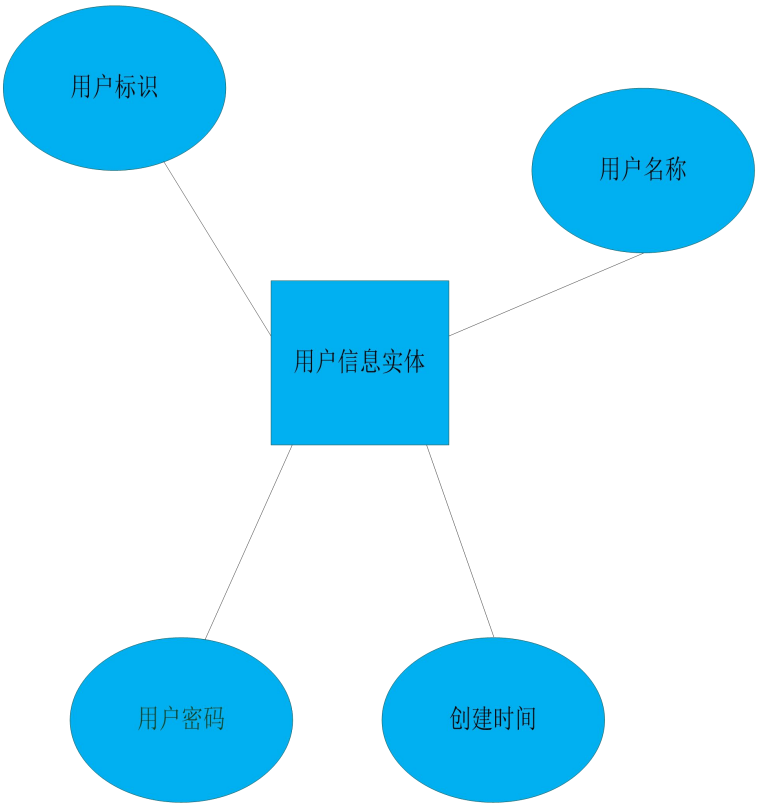
该系统中主要参与对象为元器件、型号、分系统、配套设备等数据信息，因此需要创建相应实体来保存对应信息。



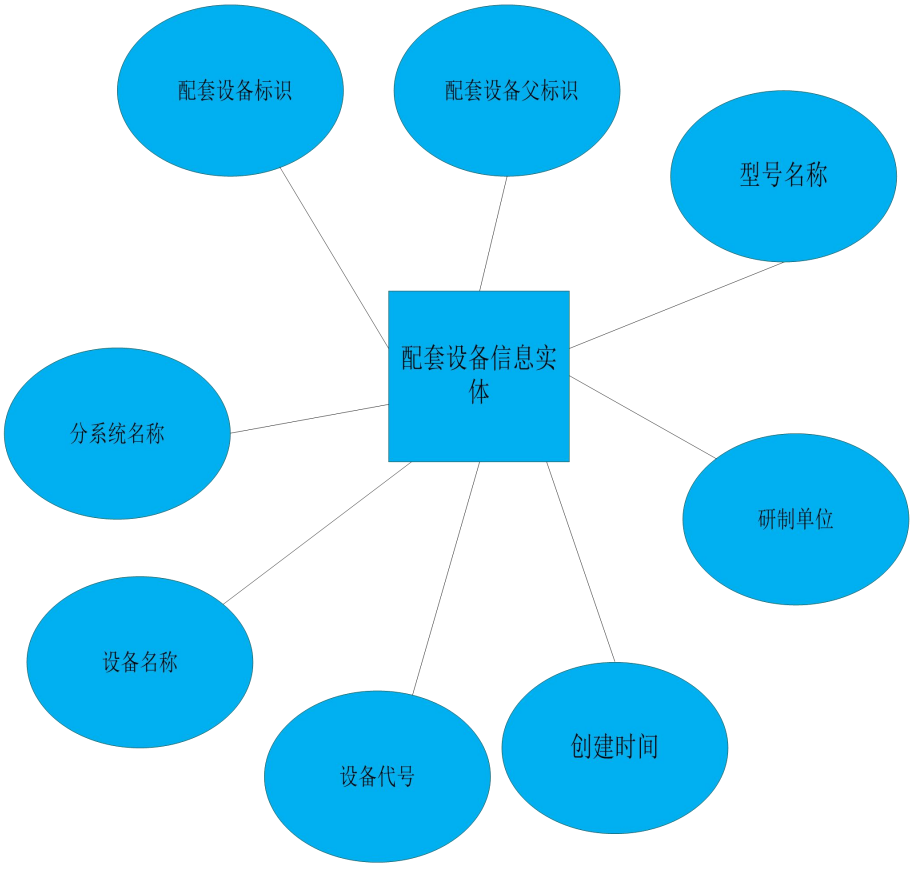
型号实体模型 E-R 图



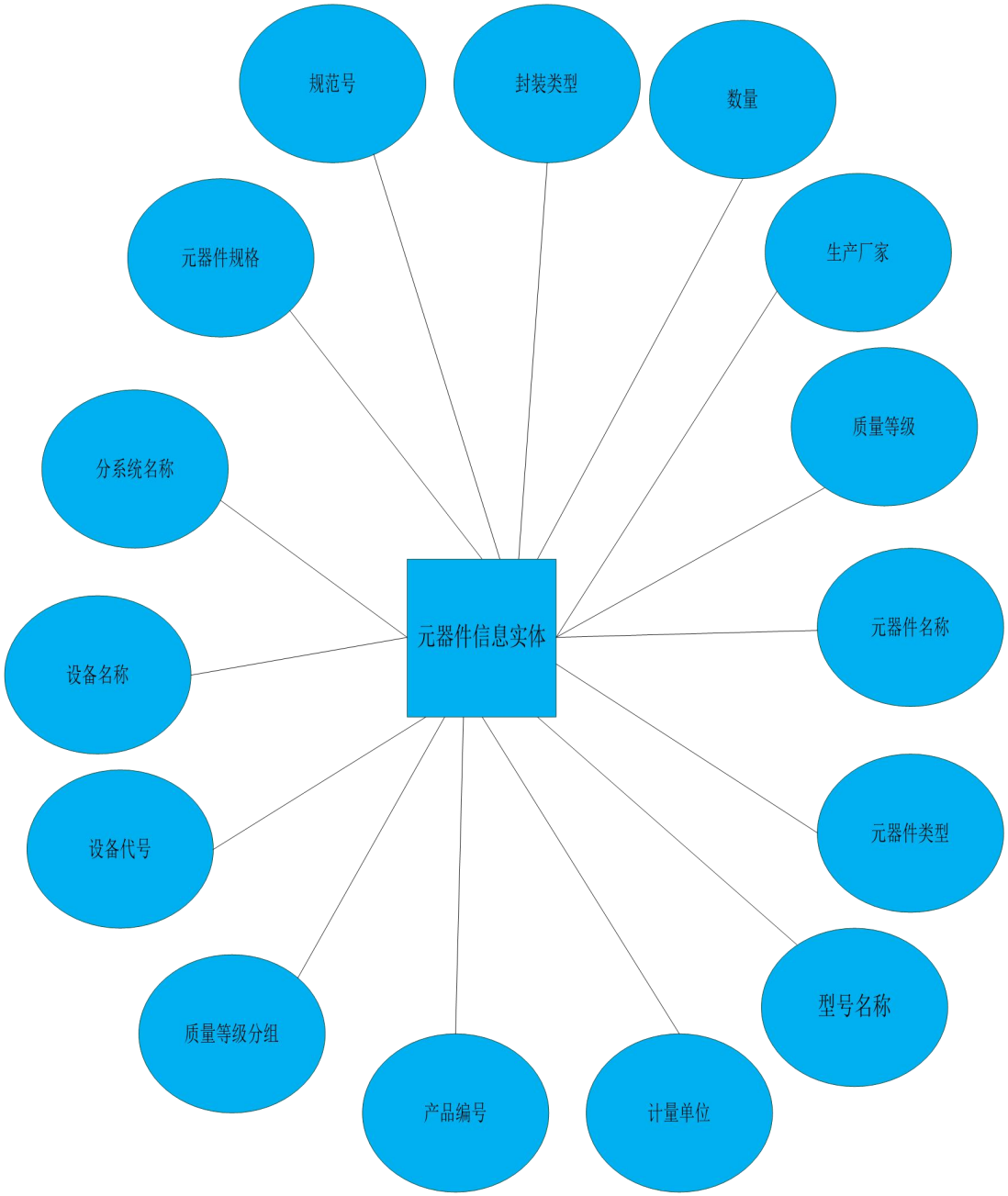
分系统实体模型 E-R 图



用户实体模型 E-R 图



配套设备实体模型 E-R 图



元器件实体模型 E-R 图

4.7.2 物理表设计

本系统的数据库管理系统为 Access2010 数据库。数据库中的表包含型号表、用户信息表、分系统表、配套设备表、元器件表。

1、型号表					
序号	数据项标识	数据项描述	类型	长度	主键

1	OBJID	型号 ID	VARCHAR2	18	P
2	XH	型号名称	VARCHAR2	32	
2、用户信息表					
序号	数据项标识	数据项描述	类型	长度	主键
1	OBJID	分系统 ID	VARCHAR2	18	P
2	SUSERNAME	用户名称	VARCHAR2	32	
3	SUSERPWD	用户密码	VARCHAR2	32	
4	CREATETIME	创建时间	Datetime	8	
3、分系统表					
序号	数据项标识	数据项描述	类型	长度	主键
1	OBJID	分系统 ID	VARCHAR2	18	P
2	XH	型号名称	VARCHAR2	32	
3	SUSERNAME	分系统名称	VARCHAR2	32	
4	CREATETIME	创建时间	Datetime	8	
4、配套设备表					
序号	数据项标识	数据项描述	类型	长度	主键
1	OBJID	设备 ID	VARCHAR2	18	P
2	XH	型号名称	VARCHAR2	32	
3	FXT_NAME	分系统名称	VARCHAR2	32	
4	SB_MC	设备名称	VARCHAR2	64	
5	SB_DH	设备代号	VARCHAR2	64	
6	SJ_DW	研制单位	VARCHAR2	64	
7	FXT_TYPE	分系统类型	VARCHAR2	64	
8	PARENT_ID	父 ID	VARCHAR2	18	
5、元器件表					
序号	数据项标识	数据项描述	类型	长度	主键
1	OBJID	元器件 ID	VARCHAR2	18	P
2	XH	型号名称	VARCHAR2	32	

3	FXT_NAME	分系统名称	VARCHAR2	32	
4	SB_MC	设备名称	VARCHAR2	64	
5	SB_DH	设备代号	VARCHAR2	64	
6	PROD_BH	产品编号	VARCHAR2	64	
7	DJ_YYDJ	单机等级	VARCHAR2	64	
8	YQJ_FL	元器件类型	VARCHAR2	18	
9	YQJ_MC	元器件名称	VARCHAR2	64	
10	XH_GG	元器件规格	VARCHAR2	64	
11	GFH	规范号	VARCHAR2	64	
12	FZ_XS	封装类型	VARCHAR2	32	
13	ZL_DJ	质量等级	VARCHAR2	32	
14	ZLDJ_FZ	质量等级分组	VARCHAR2	32	
15	JL_DW	计量单位	VARCHAR2	32	
16	DJ_YL	数量	INT	3	
17	SC_CJ	生产厂家	VARCHAR2	64	
18	SY_DW	使用单位	VARCHAR2	64	
19	GC_JK	国产 OR 国外	VARCHAR2	4	
20	SF_SCXY	是否首次选用	VARCHAR2	4	

第五章 系统界面功能实现与运行实例

5.1 型号管理

型号管理界面有两部分，上面是操作按钮，下面是型号表格。具体示例如下：

型号维护界面，具体示例如下：

5.2 用户管理

用户管理界面有两部分，上面是操作按钮，下面是分系统表格。具体示例如

下：

序号	型号	用户名	密码
1	TGTH(正样件)	AAA	0
2	TGTH(正样件)	BBB	0
3	TGTH(正样件)	CCC	0
4	TGTH(正样件)	DDD	0
5	TGTH(正样件)	对接与转位机构	0
6	TGTH(正样件)	结构机构	0

5.3 设备配套管理

设备配套界面有三部分，上边是操作按钮的工具栏，左边是分系统下的配套设备，右边为需要维护的配套设备记录。具体示例如下：

设备配套实现代码

1）初始化设备配套表格

```
//初始化设备配套表格
BOOL CDlgEquiptModify::InitGridHead(CGridCtrl &grid)
{
    int col = 0;
    GV_ITEM Item1;
    Item1.mask = GVIF_TEXT|GVIF_FORMAT;
    Item1.row = 0;
    Item1.col = col;
    Item1.nFormat =
DT_CENTER|DT_VCENTER|DT_SINGLELINE|DT_END_ELLIPSIS;

    Item1.strText.Format(_T("序号"),col);
    grid.SetColumnWidth(col,40);

    grid.SetItem(&Item1);

    col = 1;
    int nWidth = 80;
    POSITION CurPos = m_cPTCls.GetAttrList()->GetHeadPosition();
    while (CurPos)
    {
        nWidth = 80;
        CKBAttr *pAttr = (CKBAttr *)m_cPTCls.GetAttrList()->GetNext(CurPos);
        CString strAiasName    = pAttr->GetAlias();
        CString strName = pAttr->GetName();
```

```
        if(strName == "PARENT_ID" || strName == "QD_LX" || strName ==
"FXT_TYPE")
            continue;

        if( strName == "DJ_YYCSD")
            nWidth = 100;

        GV_ITEM Item;
        Item.mask = GVIF_TEXT|GVIF_FORMAT;
        Item.row = 0;
        Item.col = col;
        Item.nFormat =
DT_CENTER|DT_VCENTER|DT_SINGLELINE|DT_END_ELLIPSIS;
        grid.SetItem(&Item);

        Item.strText.Format(_T(strAiasName),col);
        grid.SetColumnWidth(col,nWidth);

        grid.SetItem(&Item);

        col++;
    }
    // fill rows/cols with text
    return TRUE;
    }
```

2）保存数据

```
    //保存
    BOOL CDlgEquiptModify::SaveData(BOOL bPrompt)
    {
        ...
            for (int nRow = 1; nRow<m_Grid.GetRowCount(); nRow++)
            {
                if (strcmp(m_Grid.GetItemText(nRow, 0), "") == 0)
                    continue;

                int nCol = 0;
                CString strRowFxtName = "";    //表格中的分系统
                POSITION pos = m_cPTCls.GetAttrList()->GetHeadPosition();
                while (pos)
                {
                    CKBAttr *pAttr = (CKBAttr
    *)m_cPTCls.GetAttrList()->GetNext(pos);
                    if (pAttr == NULL)
                        continue;
```

```
                    CString strName = pAttr->GetName();
                    CString strAlias = pAttr->GetAlias();

                    if(strName == "PARENT_ID" || strName == "QD_LX" ||
strName == "FXT_TYPE" )
                    {
                        continue;
                    }
                    nCol++;
                    CString strVal = "";

                    CGridCellDateTime *pCell = (CGridCellDateTime
*)m_Grid.GetCell(nRow, nCol);
                    if(pCell == NULL)
                    {

                    }
                    else
if(pCell->IsKindOf(RUNTIME_CLASS(CGridCellDateTime)))
                    {
                        CTime *pTime = pCell->GetTime();

                        time_t time= pTime->GetTime();

                        CString strTime;
                        strTime.Format("%d-%d-%d", pTime->GetYear(),
pTime->GetMonth(), pTime->GetDay());
                        COleDateTime cOleDate(time);
                        strTime.Format("%d-%d-%d", cOleDate.GetYear(),
cOleDate.GetMonth(), cOleDate.GetDay());

                        strVal = pCell->GetText();
                    }
                    else if(m_Grid.GetCell(nRow,
nCol)->IsKindOf(RUNTIME_CLASS(CGridCellListBox)))
                    {
                        CGridCellListBox *pCell = NULL;
                        pCell = (CGridCellListBox *)m_Grid.GetCell(nRow,
nCol);
                        CString strVal = pCell->GetText();
                    }
                    else
                    {
                        strVal = m_Grid.GetItemText(nRow, nCol);
```

```
                }

                if((strName == "FXT_NAME"       || strName == "SB_MC" )
&& strVal == "")
                {
                    CString strErrorTemp = "";
                    strErrorTemp.Format("第%d 行%s 列不应该为空
",nRow,strAlias);
                    if(strError == "")
                        strError = strErrorTemp;
                    else
                    {
                        strError += "\r\n";
                        strError += strErrorTemp;
                    }
                }
                if(strName == "FXT_NAME")
                {
                    strRowFxtName = strVal;
                }
            }//while
            if(strRowFxtName != "")
            {
                CString strFxtName =
gcCAPPGlobalEnv.GetCurrentUser()->GetUserAlias();
                if(strFxtName == "总体")
                {
                    CString sParebtObjidTemp = "";
                    m_cMapFxt.Lookup(strRowFxtName, sParebtObjidTemp);
                    if(sParebtObjidTemp != "")
                    {
                    }
                    else   //该型号下没有，分系统。
                    {
                        CString strErrorTemp = "";
                        strErrorTemp.Format("第【%d】行，【%s】型号下不
存在【%s】分系统",nRow,m_sXhName,strRowFxtName);
                        AfxMessageBox(strErrorTemp);
                        return FALSE;

                    }
                }
                else   //分系统
                {
```

```
                    if(strRowFxtName != strFxtName)
                    {
                        CString strErrorTemp = "";
                        strErrorTemp.Format("第【%d】行，【%s】型号下【%s】
分系统与当前分系统不一致！",nRow,m_sXhName,strRowFxtName);
                        AfxMessageBox(strErrorTemp);
                        return FALSE;
                    }
                }
            }
        }
    ...
}
```

5. 4 元器件管理

元器件管理界面有三部分，上面是操作按钮的工具栏，左边是分系统下的配套设备，右边为需要维护的元器件记录。具体示例如下：

元器件实现代码

1）保存元器件数据

```
BOOL CDlgPtInst::SaveData(BOOL bPrompt)
{
    ...
    for (int nRow = 1; nRow<m_Grid.GetRowCount(); nRow++)
    {
```

```
        if (strcmp(m_Grid.GetItemText(nRow, 0), "") == 0)
            continue;
        int nCol = 0;

        POSITION pos = m_cYqjCls.GetAttrList()->GetHeadPosition();
        while (pos)
        {
            CKBAttr *pAttr = (CKBAttr *)m_cYqjCls.GetAttrList()->GetNext(pos);
            if (pAttr == NULL)
                continue;

            CString strName = pAttr->GetName();
            CString strAlias = pAttr->GetAlias();
            if( strName == "QD_LX")
                continue;

            nCol++;

            CString strVal = "";

            CGridCellDateTime *pCell = (CGridCellDateTime
*)m_Grid.GetCell(nRow, nCol);
            if(pCell == NULL)
            {

            }
            else if(pCell->IsKindOf(RUNTIME_CLASS(CGridCellDateTime)))
            {
                CTime *pTime = pCell->GetTime();

                time_t time= pTime->GetTime();

                CString strTime;
                strTime.Format("%d-%d-%d", pTime->GetYear(),
pTime->GetMonth(), pTime->GetDay());
                COleDateTime cOleDate(time);
                strTime.Format("%d-%d-%d", cOleDate.GetYear(),
cOleDate.GetMonth(), cOleDate.GetDay());
                strVal = pCell->GetText();

            }
            else if(m_Grid.GetCell(nRow,
nCol)->IsKindOf(RUNTIME_CLASS(CGridCellListBox)))
            {
```

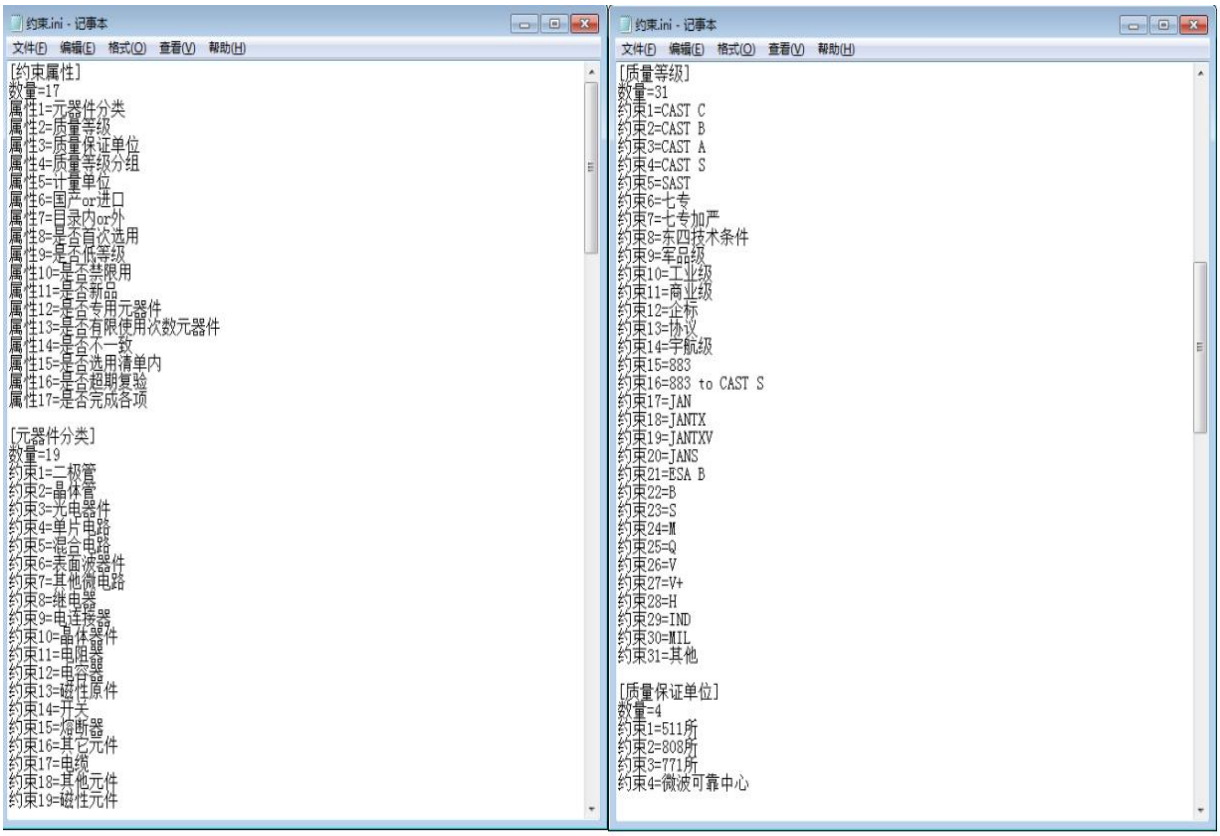
```
                CGridCellListBox *pCell = NULL;
                pCell = (CGridCellListBox *)m_Grid.GetCell(nRow, nCol);
                strVal = pCell->GetText();
            }
            else
            {
                strVal = m_Grid.GetItemText(nRow, nCol);
            }
            if((strName == "YQJ_FL"       || strName == "YQJ_MC"         ||
strName == "XH_GG" || strName == "GFH" ||
                strName == "FZ_XS"       || strName == "ZL_DJ"       || strName
== "KZJL" ||
                strName == "KDLZ_SD"         || strName == "KDLZ_FZ"     ||
strName == "JL_DW" ||
                strName == "DJ_YL"    || strName == "SC_CJ"         || strName ==
"ZL_BZDW" ||
                strName == "SF_FZYQJ"       || strName == "ML_NW"         ||
strName == "SF_SCXY" ||
                strName == "SF_DYGDZLDJ"       || strName == "GC_JK"         ||
strName == "SX_JL" ||
                strName == "SB_MC"      || strName == "SB_DH"         || strName
== "F_XT" ||
                strName == "XH" || strName == "REMARK" ) && strVal == "")
             {
                CString strErrorTemp = "";
                strErrorTemp.Format("第【%d】行【%s】列不应该为空
",nRow,strAlias);
                if(strError == "")
                    strError = strErrorTemp;
                else
                {
                    strError += "\r\n";
                    strError += strErrorTemp;
                }
            }

            CConstraint *pConstraint = NULL;

    theApp.m_cMapConstraintSelect.Lookup(strAlias,(CObject*&)pConstraint);
            if (pConstraint != NULL)
            {
                BOOL bFind = FALSE;
                for(int i = 0; i < pConstraint->m_cConstraintArray.GetSize(); i++)
                {
```

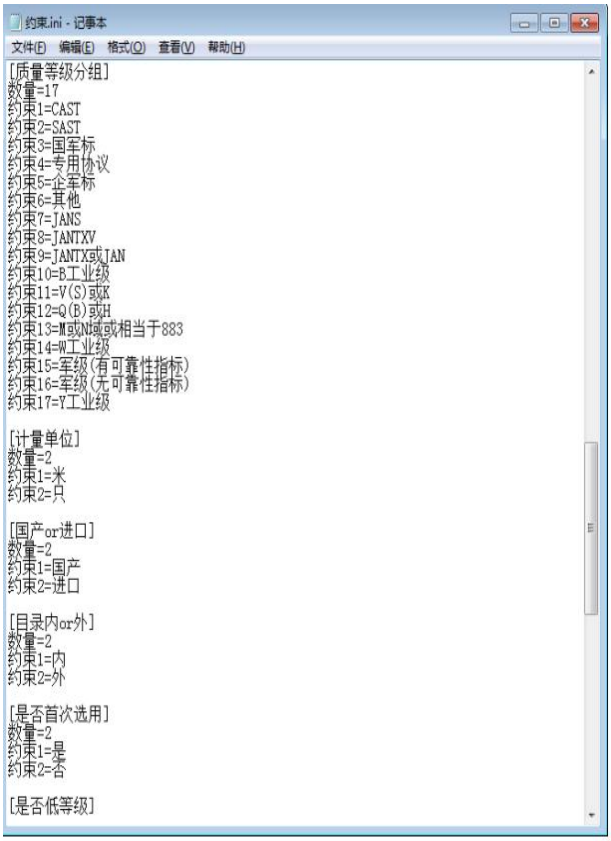
```
                    CString strYqjFlTemp =
pConstraint->m_cConstraintArray.GetAt(i);
                    if(strYqjFlTemp == strVal)
                    {
                        bFind = TRUE;
                        break;
                    }
                }
                if(!bFind)
                {
                    CString strErrorTemp = "";
                    strErrorTemp.Format("第【%d】行【%s】列的值不符合要求！
",nRow,strAlias);
                    if(strError == "")
                        strError = strErrorTemp;
                    else
                    {
                        strError += "\r\n";
                        strError += strErrorTemp;
                    }
                }
            }

        #endif

            if(strName == "DJ_YL")
            {
                if(!CheckStringIsInt(strVal))//不全为数字
                {
                    CString strErrorTemp = "";
                    strErrorTemp.Format("第【%d】行【%s】列的值不全为数字！
",nRow,strAlias);
                    if(strError == "")
                        strError = strErrorTemp;
                    else
                    {
                        strError += "\r\n";
                        strError += strErrorTemp;
                    }
                }
            }
        }

        ...
```

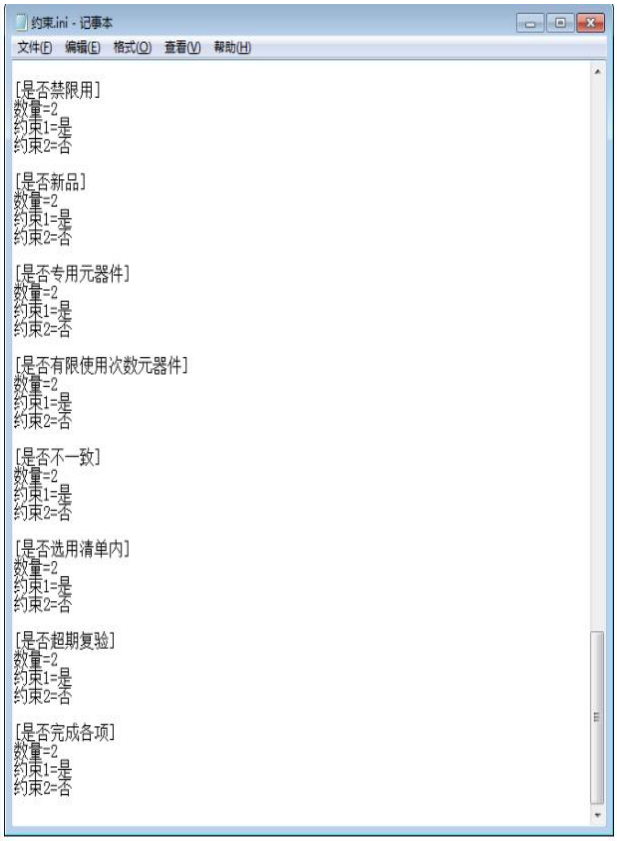
}

5.5 配置管理

在元器件录入过程中，有些属性通过下拉框实现的，下拉框内容是通过当前应用程序目录下[约束.ini]文件进行配置的。部分配置示例如下：



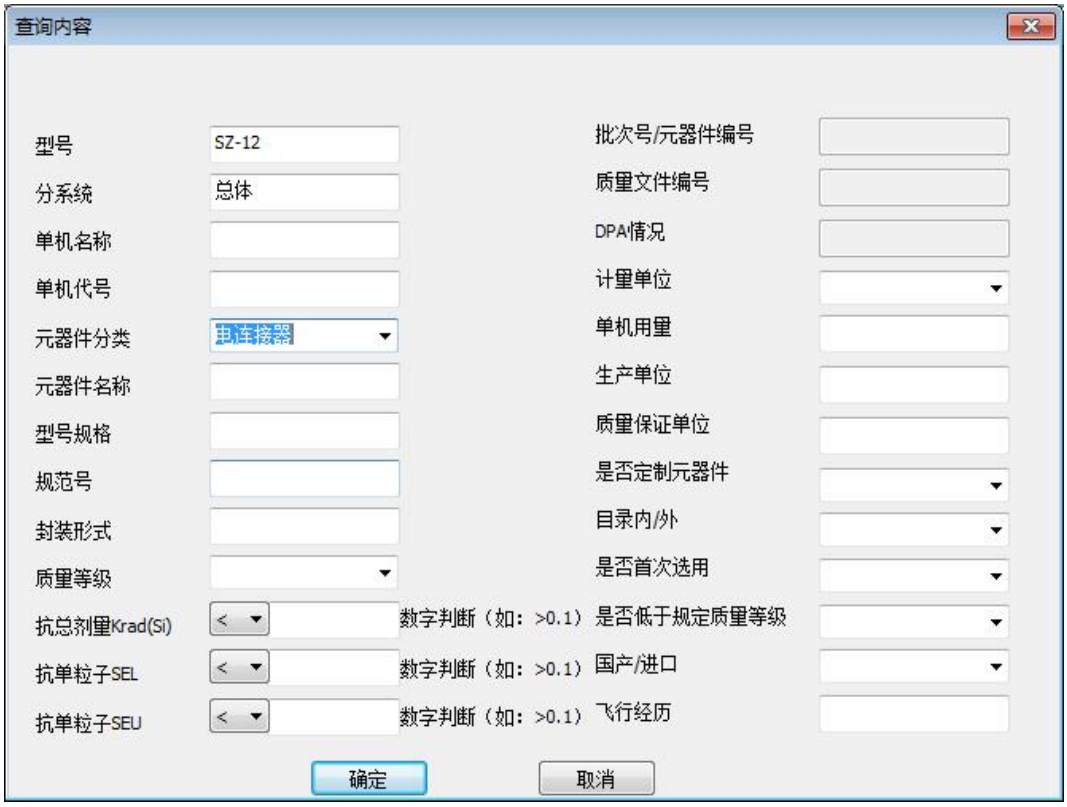


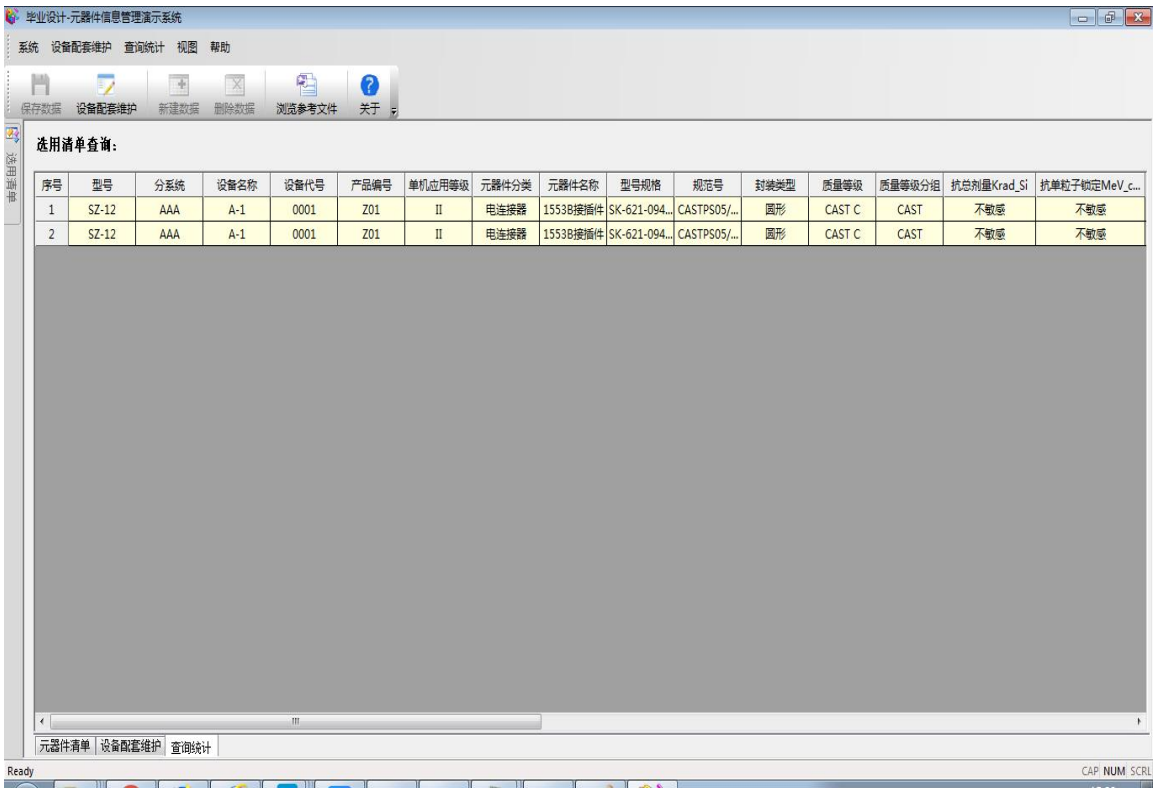


5.6 查询统计管理

5.6.1 通用查询

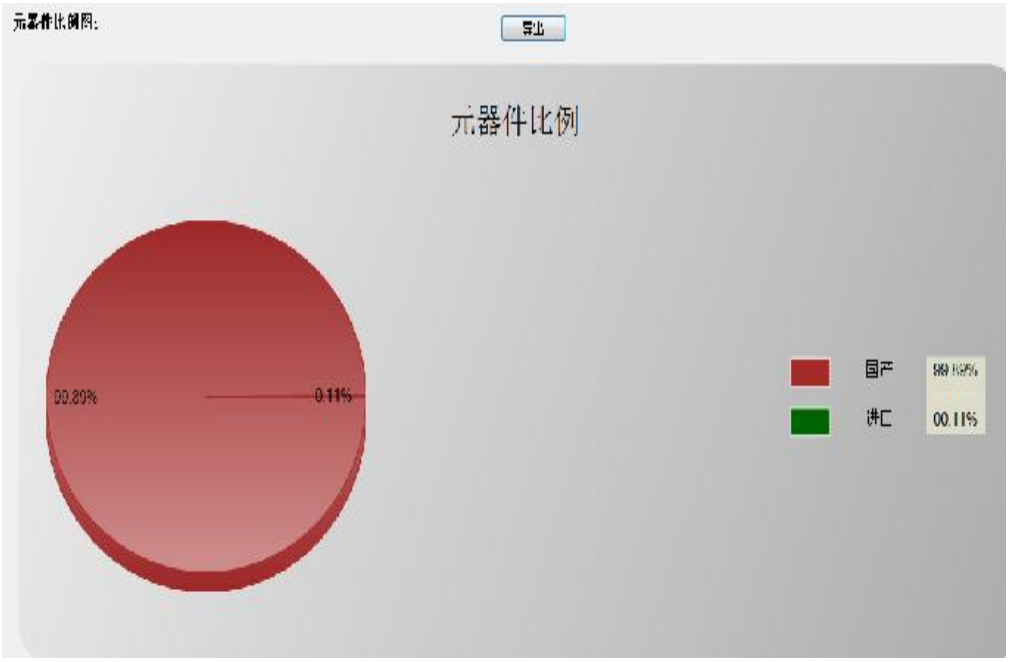
通过通用查询功能可在查询结果显示在查询列表中。具体示例如下：





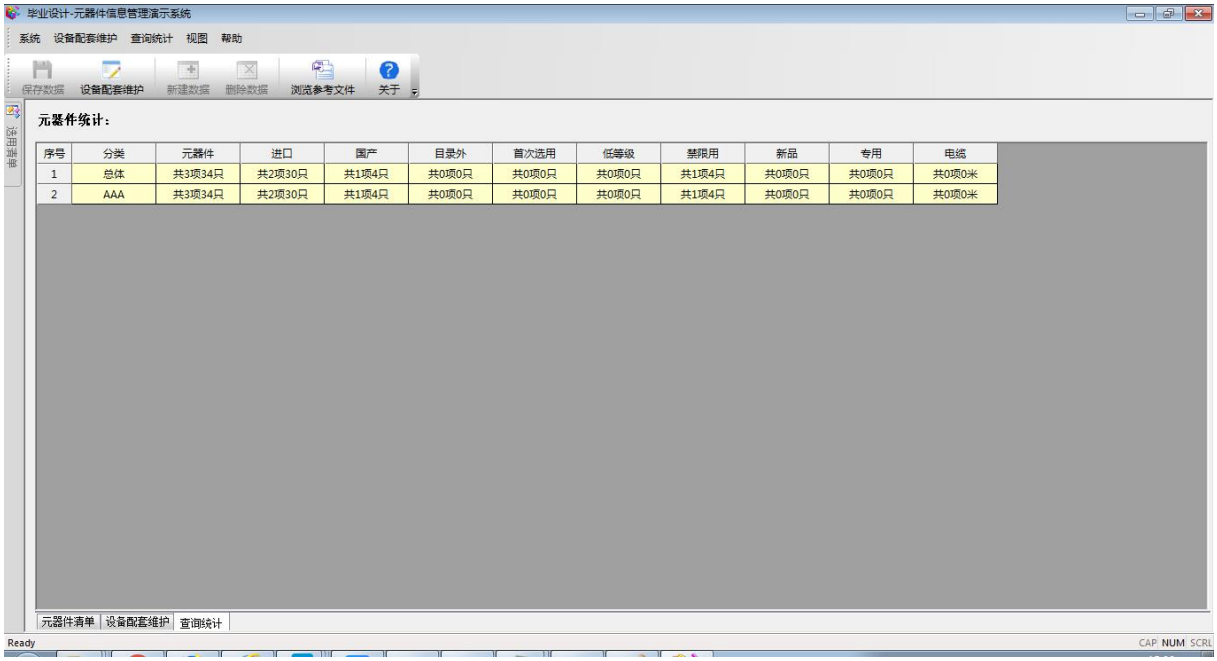
5.6.2 元器件比例汇总

用饼图显示某型号下的元器件国产、国外所占比例。具体示例如下：



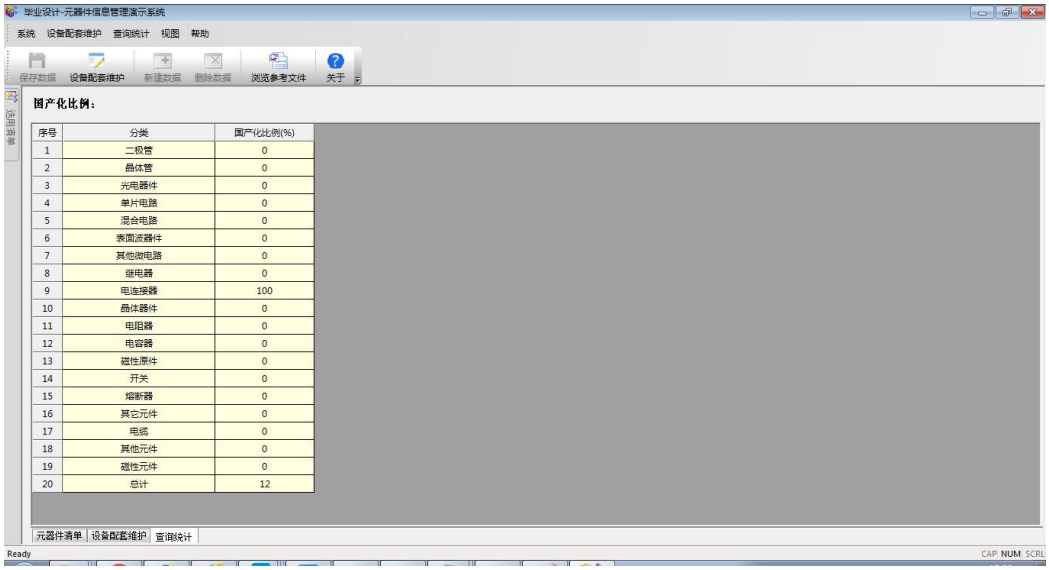
5.6.3 元器件统计

用列表显示某型号下的不同分类的元器件总数。具体示例如下：



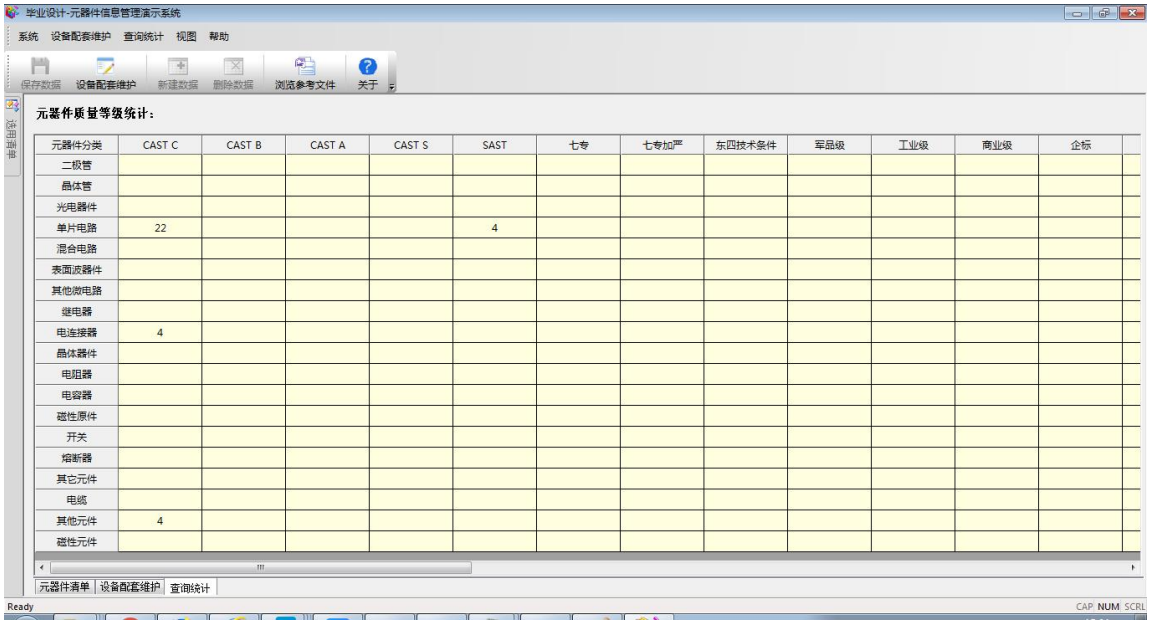
5.6.4 国产化比例

用列表显示某型号下的不同分类的元器件国产化比例。具体示例如下：



5.6.5 质量等级统计

用列表显示某型号下的不同分类的元器件的质量等级。具体示例如下：



5.7 系统的测试

5.7.1 系统测试

本次测试采用功能测试、性能测试、强度测试、安全测试、恢复测试、可用性测试以及安装/卸载测试。

1）功能测试。功能测试，又称黑盒测试，按照需求编写出来的测试用例，

输入数据在预期结果和实际结果之间进行评测，进而提出更加使产品达到用户使用的要求。本次功能测试主要是手工测试。

2）性能测试。测试软件的运行性能。结合强度测试，对软件性能指标，传输连接的最长时限、传输错误率、计算精度、记录精度、响应的时限和恢复时限，测试满足开发要求，能够提供设计所描述的功能。

3）强度测试。新建五个以上产品型号，每个产品型号下新增各类不同的配件，并大量重复、输入大量的数据，输入大数值数据，输入各行业不同类型名称的元器件，重复进行输入、删除、修改及保存。对录入数据进行自定义规则查询，对查询结果进行记录，确保查询准确。

4）安全测试。验证安装在系统内的保护机构确实能够对系统进行保护，使之不受各种非常的干扰。安全测试时需要设计一些测试用例试图突破系统的安全保密措施，检验系统是否有安全保密的漏洞。

5）恢复测试。采用人工的干扰使软件出错，中断使用，检测系统的恢复能力，特别是通讯系统。恢复测试时，应该参考性能测试的相关测试指标。

6）可用性测试。测试用户是否能够满意使用。具体体现为操作是否方便，用户界面是否友好等。

7）安装/卸载测试（install/uninstall test）等。

系统测试需要对被测的软件结合需求分析做仔细的测试分析，建立测试用例。

结　论

通过本次软件设计开发,让我对项目管理以及软件开发有了新的理解和认知，软件开发，首先要调研企业或目标客户目前情况，深入了解客户生产管理现实问题，以及希望通过软件解决的当前困难。只有彻底搞清楚了需求，做出来的产品和软件才是好的应用程序。

满足企业需求，永远是软件开发人员紧紧围绕的主题，要及时调整和适应需求的变化，通过我在企业工作的经验，以及企业中各部门工作性质，每个部门和人员都有自己的核心需求点，我们设计的软件只有满足每个部门人员的应用要求。这有的应用才可以真正为起来带来经济效益。

目前，计算机普及率很高，在各行业应用都比较广泛，电脑办公已非常普遍，设计的电子元器件管理系统具有良好的扩展性,可应用于各类相似管理模式的企业应用，软件简单易上手，可以帮助微小企业进行物料管理。

元器件管理系统基础功能模块已建立，后续希望增加更多的功能模块；加入更多可扩展功能；优化人机交互界面；不断完善各类企业需要的个性化功能；可以建立专家库，例如新增一个型号时候可以对其进行定制相应的配套系统。可以根据企业规模和性质，建立各行业的元器件管理系统；提高软件运行稳定性。让软件更好的为使用者服务。也希望目前系统的功能几乎可以适应各类制造企业元器件管理使用。

参考文献

[1] 高勇，符敢为, GAOYong, et al. 基于 VC+ADO+Access 的数据库访问技术在 Pro/E 环境下的实现[J]. 机械设计与制造工程, 2013, 42(3):14-17.

[2] 明日科技 编. Visual C++从入门到精通（第 5 版）,北京：清华大学出版社，2019.

[3] 李春磊. 电子配件成品库存管理系统的设计与实现[D]. 2016.

[4] 基于 VS2010 的 PLC 程序编辑系统的研究与设计. Diss. 广西科技大学, 2015..

[5]谷军霞, 刘然. 基于 VC 和 ACCESS 的库存管理系统设计与实现[C] 中国气象学会气象通信与信息技术委员会暨国家气象信息中心科技年会. 2011.

[6] 贺伟,李凤. 基于项目驱动式教学的《Java 面向对象程序设计》课程实践[J]. 计算机产品与流通,2019(01):253-265..

[7]张永强. 计算机软件 Java 编程特点及其技术分析[J]. 计算机产品与流通,2019(01):23.

[8] 严竞雄. 基于 ASP.NET 的高校辅导员工作管理系统的设计与实现[J]. 电脑知识与技术,2017,13(36):56-58.

[9] 郭振勇. ASP 中实现 Excel 和 Access 数据互通[J].福建电脑, 2016, 1 (1) :88-89.

[10]王莹。浅谈大数据时代对经济学的一点思考[J].中华少年，2017,（28）：294.

[11]王祥顺。软件工程的安全检测和维护[J].电子技术与软件工程，2017,（19）：36.

[12]王涛。软件工程化的基本形式和关键技术[J].电子技术与软件工程，2017,（19）：37.

[13]叶伟。软件开发技术在软件工程管理中的应用[J].电子技术与软件工程，2017,（18）：60-61.

[14]常文光, 刘砚, 范收平. 基于主数据管理的电子元器件基础信息管理系统[J]. 科技视界, 2018, No.238(16):36+56-57.

[15]王明月;企业知识文档检索管理系统的设计与实现[D];哈尔滨工业大学;2017年

[16]杜建平, 潘大程. 一种电子元器件试验数据管理系统的设计和优化研究[J].

信息化建设, 2016(7).

[17]明日科技 编. Visual C++从入门到精通（第 5 版）,北京：清华大学出版社，2019.

[18]冯晓星，马晓静. VC 基于 ADO 技术访问 Access 数据库[J]. 计算机与网络, 2013(08):56-58.

致　谢

我是04年毕业走上社会，开始工作，由于工作性质的原因，经常会深入各类企业做配合服务，在这个过程中，我对各类企业也有了一个了解，也希望通过学习更好的做好自己的工作，也希望进一步努力提升自己。通过各方面的了解以及自己的实际走访，我决定选择国家开放大学来继续深造，国家开放大学的理念及教学模式，能让我边工作，边学习，通过学习提高了我的工作水平，工作的经验及方法也让我的学习效果更好。

在国家开放大学学习期间，我看到了来自各行业的优秀人员还在继续学习，也看到很多严谨认真的老师在传授知识，努力工作，我的第一体会就是任何时候，我们都不能忘记学习，在学校学习期间非常感谢各科老师的关怀与鼓励，感谢他妈对我的指导和监督。

在毕业设计和论文完成，我要特别感谢感谢我的论文指导老师蒋漪涟老师，她是一位非常热情，负责认真的老师，在前期软件设计的时候，我们进行了多次的沟通，从需求的调研及对需求的分析、建模的过程和实现方式上都给我提出了宝贵的意见。在毕业论文的撰写方面，他都认真的对我进行了指导和详细的修改，从和她的交流沟通中，我也深受感动和鼓舞。蒋老师一丝不苟，严谨认真的教学作风和深厚的理论水平，启迪我在未来的工作和学习中不忘初心，砥砺前行。我要向指导老师表达我内心最崇高的敬意和最衷心的感谢。

最后再次感谢国家开放大学给我这次继续学习的机会，国家开放大学也将在我们这些学子身上烙印，这段学习生活的时光将影响我的一生，我也将代表国家开放大学毕业生，铭记学校和老师的指导，以一百倍的信心和万分的努力工作回报社会汇报母校。

谢谢！

附录 1

MainFrm.h

```
// MainFrm.h : interface of the CMainFrame class
//
#pragma once
#include "ClassView.h"
#include "OutputWnd.h"
class CMainFrame : public CFrameWndEx
{
protected: // create from serialization only
	CMainFrame();
	DECLARE_DYNCREATE(CMainFrame)
// Attributes
// Operations
public:
// Overrides
public:
	virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
	virtual BOOL LoadFrame(UINT nIDResource, DWORD dwDefaultStyle =
WS_OVERLAPPEDWINDOW | FWS_ADDTOTITLE, CWnd* pParentWnd =
NULL, CCreateContext* pContext = NULL);
// Implementation
public:
	virtual ~CMainFrame();
#ifdef _DEBUG
	virtual void AssertValid() const;
	virtual void Dump(CDumpContext& dc) const;
#endif
public:
	//功能：获取类视图
	CClassView *GetClsView() ;
	void OnMenuRefreshTree();
	void OnMenuClassTreeFocus();
	void OnMenuOutTreeFocus();
	void WriteLog(CString m_sLogFilePath, CString sInfo);
	//生成试图脚本
	CMapStringToOb m_cMapViewList;//待生成视图的对象链表
	BOOL m_bSaveFlag;
protected:  // control bar embedded members
	CMFCMenuBar       m_wndMenuBar;
	CMFCToolBar       m_wndToolBar;
	CMFCStatusBar     m_wndStatusBar;
	CMFCToolBarImages m_UserImages;
```

```
    CClassView        m_wndClassView;
    CMFCToolBar       m_wndSysToolBar;
// Generated message map functions
protected:
    afx_msg int OnCreate(LPCREATESTRUCT lpCreateStruct);
    afx_msg void OnViewCustomize();
    afx_msg LRESULT OnToolbarCreateNew(WPARAM wp, LPARAM lp);
    afx_msg void OnApplicationLook(UINT id);
    afx_msg void OnUpdateApplicationLook(CCmdUI* pCmdUI);
    DECLARE_MESSAGE_MAP()
    BOOL CreateDockingWindows();
    void SetDockingWindowIcons(BOOL bHiColorIcons);
public:
    afx_msg void OnMenuEnvset();
    afx_msg void OnMenuSave();
    afx_msg void OnUpdateMenuEnvset(CCmdUI *pCmdUI);
    afx_msg void OnUpdateMenuSave(CCmdUI *pCmdUI);
    afx_msg void OnClose();
    afx_msg void OnMenuConfigView();
    afx_msg void OnUpdateMenuConfigView(CCmdUI *pCmdUI);
    afx_msg void OnMenuFind();
    afx_msg void OnUpdateMenuFind(CCmdUI *pCmdUI);
    afx_msg void OnMenuSysConfig();
    afx_msg void OnUpdateMenuSysConfig(CCmdUI *pCmdUI);
    afx_msg void OnMenuSaveYqyb();
    afx_msg void OnUpdateMenuSaveYqyb(CCmdUI *pCmdUI);
    afx_msg void OnMenuEquiptWh(); //设备配套维护
};
```

MainFrm.cpp

```
// MainFrm.cpp : implementation of the CMainFrame class
//
#include "stdafx.h"
#include "Components.h"
#include "MainFrm.h"
#include "Components.h"
#include "ComponentsView.h"
#include "DlgSysConfig.h"
#include <direct.h>
#include "Components_imp.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
// CMainFrame
IMPLEMENT_DYNCREATE(CMainFrame, CFrameWndEx)
```

```
const int    iMaxUserToolbars = 10;
const UINT uiFirstUserToolBarId = AFX_IDW_CONTROLBAR_FIRST + 40;
const UINT uiLastUserToolBarId = uiFirstUserToolBarId + iMaxUserToolbars - 1;
BEGIN_MESSAGE_MAP(CMainFrame, CFrameWndEx)
    ON_WM_CREATE()
    ON_COMMAND(ID_VIEW_CUSTOMIZE,
&CMainFrame::OnViewCustomize)
    ON_REGISTERED_MESSAGE(AFX_WM_CREATETOOLBAR,
&CMainFrame::OnToolbarCreateNew)
    ON_COMMAND_RANGE(ID_VIEW_APPLOOK_WIN_2000,
ID_VIEW_APPLOOK_OFF_2007_AQUA, &CMainFrame::OnApplicationLook)
    ON_UPDATE_COMMAND_UI_RANGE(ID_VIEW_APPLOOK_WIN_2000,
ID_VIEW_APPLOOK_OFF_2007_AQUA,
&CMainFrame::OnUpdateApplicationLook)
    ON_COMMAND(ID_MENU_SAVE, &CMainFrame::OnMenuSave)
    ON_UPDATE_COMMAND_UI(ID_MENU_ENVSET,
&CMainFrame::OnUpdateMenuEnvset)
    ON_UPDATE_COMMAND_UI(ID_MENU_SAVE,
&CMainFrame::OnUpdateMenuSave)
    ON_WM_CLOSE()
    ON_COMMAND(ID_MENU_EQUIPPT, &CMainFrame::OnMenuEquiptWh)
    ON_COMMAND(ID_MENU_ENVSET,&CMainFrame::OnMenuEnvset)
    ON_COMMAND(ID_MENU_CONFIG_VIEW,
&CMainFrame::OnMenuConfigView)
    ON_UPDATE_COMMAND_UI(ID_MENU_CONFIG_VIEW,
&CMainFrame::OnUpdateMenuConfigView)


ON_COMMAND(ID_MENU_FIND, &CMainFrame::OnMenuFind)
    ON_UPDATE_COMMAND_UI(ID_MENU_FIND,
&CMainFrame::OnUpdateMenuFind)
    ON_COMMAND(ID_MENU_SYS_CONFIG,
&CMainFrame::OnMenuSysConfig)
    ON_UPDATE_COMMAND_UI(ID_MENU_SYS_CONFIG,
&CMainFrame::OnUpdateMenuSysConfig)


    ON_COMMAND(ID_MENU_SAVE_YQYB,
&CMainFrame::OnMenuSaveYqyb)
    ON_UPDATE_COMMAND_UI(ID_MENU_SAVE_YQYB,
&CMainFrame::OnUpdateMenuSaveYqyb)
END_MESSAGE_MAP()
static UINT indicators[] =
{
```

```
    ID_SEPARATOR,           // status line indicator
    ID_INDICATOR_CAPS,
    ID_INDICATOR_NUM,
    ID_INDICATOR_SCRL,
};
// CMainFrame construction/destruction
CMainFrame::CMainFrame()
{
    // TODO: add member initialization code here
    theApp.m_nAppLook = theApp.GetInt(_T("ApplicationLook"),
ID_VIEW_APPLOOK_VS_2005);
    m_bSaveFlag = TRUE;
}
CMainFrame::~CMainFrame()
{
}
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CFrameWndEx::OnCreate(lpCreateStruct) == -1)
        return -1;
    BOOL bNameValid;
    // set the visual manager and style based on persisted value
    OnApplicationLook(theApp.m_nAppLook);
    if (!m_wndMenuBar.Create(this))
    {
        TRACE0("Failed to create menubar\n");
        return -1;          // fail to create
    }
    m_wndMenuBar.SetPaneStyle(m_wndMenuBar.GetPaneStyle() |
CBRS_SIZE_DYNAMIC | CBRS_TOOLTIPS | CBRS_FLYBY);
    // prevent the menu bar from taking the focus on activation
    CMFCPopupMenu::SetForceMenuFocus(FALSE);
    if (!m_wndToolBar.CreateEx(this, TBSTYLE_FLAT, WS_CHILD |
WS_VISIBLE | CBRS_TOP | CBRS_GRIPPER | CBRS_TOOLTIPS | CBRS_FLYBY
| CBRS_SIZE_DYNAMIC) ||
        !m_wndToolBar.LoadToolBar(theApp.m_bHiColorIcons ?
IDR_MAINFRAME : IDR_MAINFRAME))
    {
        TRACE0("未能创建工具栏\n");
        return -1;          //  未能创建
    }
    CString strToolBarName;
    bNameValid = strToolBarName.LoadString(IDS_TOOLBAR_STANDARD);
    ASSERT(bNameValid);
```

```
    m_wndToolBar.SetWindowText(strToolBarName);
    CString strCustomize;
    bNameValid = strCustomize.LoadString(IDS_TOOLBAR_CUSTOMIZE);
    ASSERT(bNameValid);
    m_wndToolBar.EnableCustomizeButton(TRUE, ID_VIEW_CUSTOMIZE,
strCustomize);
    // Allow user-defined toolbars operations:
    InitUserToolbars(NULL, uiFirstUserToolBarId, uiLastUserToolBarId);
    if (!m_wndStatusBar.Create(this))
    {
        TRACE0("Failed to create status bar\n");
        return -1;          // fail to create
    }
    m_wndStatusBar.SetIndicators(indicators, sizeof(indicators)/sizeof(UINT));
    // TODO: Delete these five lines if you don't want the toolbar and menubar to be
dockable
    m_wndMenuBar.EnableDocking(CBRS_ALIGN_ANY);
    m_wndToolBar.EnableDocking(CBRS_ALIGN_ANY);
    EnableDocking(CBRS_ALIGN_ANY);
    DockPane(&m_wndMenuBar);
    DockPane(&m_wndToolBar);
    m_wndToolBar.EnableTextLabels(TRUE);
    // enable Visual Studio 2005 style docking window behavior
    CDockingManager::SetDockingMode(DT_SMART);
    // enable Visual Studio 2005 style docking window auto-hide behavior
    EnableAutoHidePanes(CBRS_ALIGN_ANY);
    // Load menu item image (not placed on any standard toolbars):
    // create docking windows
    if (!CreateDockingWindows())
    {
        TRACE0("Failed to create docking windows\n");
        return -1;
    }
    //m_wndFileView.EnableDocking(CBRS_ALIGN_ANY);
    m_wndClassView.EnableDocking(CBRS_ALIGN_ANY);
    DockPane(&m_wndClassView);
    // 启用工具栏和停靠窗口菜单替换
    EnablePaneMenu(TRUE, ID_VIEW_CUSTOMIZE, strCustomize,
ID_VIEW_TOOLBAR);
    // 启用快速(按住 Alt 拖动)工具栏自定义
    CMFCToolBar::EnableQuickCustomization();
    return 0;
}
BOOL CMainFrame::PreCreateWindow(CREATESTRUCT& cs)
```

```
{
    if( !CFrameWndEx::PreCreateWindow(cs) )
        return FALSE;
    // TODO: Modify the Window class or styles here by modifying
    //    the CREATESTRUCT cs
    cs.style = WS_OVERLAPPED | WS_CAPTION | FWS_ADDTOTITLE
         | WS_THICKFRAME | WS_MINIMIZEBOX | WS_MAXIMIZEBOX |
WS_MAXIMIZE | WS_SYSMENU;
    cs.style&=~FWS_ADDTOTITLE;
    cs.lpszName= "毕业设计-元器件信息管理演示系统";
    cs.lpszClass=AfxRegisterWndClass(CS_HREDRAW | CS_VREDRAW,0,0,
        AfxGetApp()->LoadIcon(IDR_MAINFRAME));
    return TRUE;
}
BOOL CMainFrame::CreateDockingWindows()
{
    BOOL bNameValid;
    // Create class view
    CString strClassView;
    bNameValid = strClassView.LoadString(IDS_CLASS_VIEW);
    ASSERT(bNameValid);
    if (!m_wndClassView.Create(strClassView, this, CRect(0, 0, 200, 200), TRUE,
ID_VIEW_CLASSVIEW, WS_CHILD | WS_VISIBLE | WS_CLIPSIBLINGS |
WS_CLIPCHILDREN | CBRS_LEFT | CBRS_FLOAT_MULTI))
    {
        TRACE0("Failed to create Class View window\n");
        return FALSE; // failed to create
    }
    SetDockingWindowIcons(theApp.m_bHiColorIcons);
    return TRUE;
}
void CMainFrame::SetDockingWindowIcons(BOOL bHiColorIcons)
{
    HICON hClassViewIcon = (HICON) ::LoadImage(::AfxGetResourceHandle(),
MAKEINTRESOURCE(bHiColorIcons ? IDI_CLASS_VIEW_HC :
IDI_CLASS_VIEW),
IMAGE_ICON, ::GetSystemMetrics(SM_CXSMICON), ::GetSystemMetrics(SM_C
YSMICON), 0);
    m_wndClassView.SetIcon(hClassViewIcon, FALSE);
}
// CMainFrame diagnostics
#ifdef _DEBUG
void CMainFrame::AssertValid() const
{
```

```
    CFrameWndEx::AssertValid();
}
void CMainFrame::Dump(CDumpContext& dc) const
{
    CFrameWndEx::Dump(dc);
}
#endif //_DEBUG
// CMainFrame message handlers
void CMainFrame::OnViewCustomize()
{
    CMFCToolBarsCustomizeDialog* pDlgCust = new
CMFCToolBarsCustomizeDialog(this, TRUE /* scan menus */);
    pDlgCust->EnableUserDefinedToolbars();
    pDlgCust->Create();
}
LRESULT CMainFrame::OnToolbarCreateNew(WPARAM wp,LPARAM lp)
{
    LRESULT lres = CFrameWndEx::OnToolbarCreateNew(wp,lp);
    if (lres == 0)
    {
        return 0;
    }
    CMFCToolBar* pUserToolbar = (CMFCToolBar*)lres;
    ASSERT_VALID(pUserToolbar);
    BOOL bNameValid;
    CString strCustomize;
    bNameValid = strCustomize.LoadString(IDS_TOOLBAR_CUSTOMIZE);
    ASSERT(bNameValid);
    pUserToolbar->EnableCustomizeButton(TRUE, ID_VIEW_CUSTOMIZE,
strCustomize);
    return lres;
}
void CMainFrame::OnApplicationLook(UINT id)
{
    CWaitCursor wait;
    theApp.m_nAppLook = id;
    switch (theApp.m_nAppLook)
    {
    case ID_VIEW_APPLOOK_WIN_2000:
    CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFCVisualMa
nager));
        break;
    case ID_VIEW_APPLOOK_OFF_XP:
```

```
    CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFCVisualMa
nagerOfficeXP));
        break;
    case ID_VIEW_APPLOOK_WIN_XP:
        CMFCVisualManagerWindows::m_b3DTabsXPTheme = TRUE;

    CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFCVisualMa
nagerWindows));
        break;
    case ID_VIEW_APPLOOK_OFF_2003:

    CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFCVisualMa
nagerOffice2003));
        CDockingManager::SetDockingMode(DT_SMART);
        break;
    case ID_VIEW_APPLOOK_VS_2005:

    CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFCVisualMa
nagerVS2005));
        CDockingManager::SetDockingMode(DT_SMART);
        break;
    default:
        switch (theApp.m_nAppLook)
        {
        case ID_VIEW_APPLOOK_OFF_2007_BLUE:

    CMFCVisualManagerOffice2007::SetStyle(CMFCVisualManagerOffice2007::Of
fice2007_LunaBlue);
            break;
        case ID_VIEW_APPLOOK_OFF_2007_BLACK:

    CMFCVisualManagerOffice2007::SetStyle(CMFCVisualManagerOffice2007::Of
fice2007_ObsidianBlack);
            break;
        case ID_VIEW_APPLOOK_OFF_2007_SILVER:

    CMFCVisualManagerOffice2007::SetStyle(CMFCVisualManagerOffice2007::Of
fice2007_Silver);
            break;
        case ID_VIEW_APPLOOK_OFF_2007_AQUA:

    CMFCVisualManagerOffice2007::SetStyle(CMFCVisualManagerOffice2007::Of
fice2007_Aqua);
            break;
```

```
        }

    CMFCVisualManager::SetDefaultManager(RUNTIME_CLASS(CMFCVisualMa
nagerOffice2007));
        CDockingManager::SetDockingMode(DT_SMART);
    }
    RedrawWindow(NULL, NULL, RDW_ALLCHILDREN | RDW_INVALIDATE
| RDW_UPDATENOW | RDW_FRAME | RDW_ERASE);
    theApp.WriteInt(_T("ApplicationLook"), theApp.m_nAppLook);
}
void CMainFrame::OnUpdateApplicationLook(CCmdUI* pCmdUI)
{
    pCmdUI->SetRadio(theApp.m_nAppLook == pCmdUI->m_nID);
}
BOOL CMainFrame::LoadFrame(UINT nIDResource, DWORD dwDefaultStyle,
CWnd* pParentWnd, CCreateContext* pContext)
{
    // base class does the real work
    if (!CFrameWndEx::LoadFrame(nIDResource, dwDefaultStyle, pParentWnd,
pContext))
    {
        return FALSE;
    }
    // enable customization button for all user toolbars
    BOOL bNameValid;
    CString strCustomize;
    bNameValid = strCustomize.LoadString(IDS_TOOLBAR_CUSTOMIZE);
    ASSERT(bNameValid);
    for (int i = 0; i < iMaxUserToolbars; i ++)
    {
        CMFCToolBar* pUserToolbar = GetUserToolBarByIndex(i);
        if (pUserToolbar != NULL)
        {
            pUserToolbar->EnableCustomizeButton(TRUE,
ID_VIEW_CUSTOMIZE, strCustomize);
        }
    }
    return TRUE;
}
void CMainFrame::OnMenuEnvset()
{
}
void CMainFrame::OnMenuSave()//对当前类信息进行保存
{
```

```
    if(gpSIMView == NULL)
        return ;
    //保存数据
    gpSIMView->SaveData();
}
CClassView *CMainFrame::GetClsView()
{
    return &m_wndClassView;
}
void CMainFrame::OnUpdateMenuEnvset(CCmdUI *pCmdUI)
{
    if(!gIsEditFlag)
    {
        pCmdUI->Enable(FALSE);
        return ;
    }
    pCmdUI->Enable(TRUE);
}
void CMainFrame::OnClose()
{
    // TODO: Add your message handler code here and/or call default
    SetFocus();
    CComponentsView *pView = (CComponentsView*)GetActiveView();
    if (pView != NULL)
    {
        if (pView->IsModefied())
        {
            int nRet = AfxMessageBox("实例已发生更改,是否保存?",
MB_YESNO);
            if (nRet == IDYES)
            {
                BOOL bSaveFlag = pView->SaveData();
                if(!bSaveFlag)
                    return;
            }
        }
    }

    CFrameWndEx::OnClose();
}
void CMainFrame::OnUpdateMenuUpdatedb(CCmdUI *pCmdUI)
{
    // TODO: Add your command update UI handler code here
    if(!gIsEditFlag)
```

```
    {
        pCmdUI->Enable(FALSE);
        return ;
    }
    pCmdUI->Enable(TRUE);
}
void CMainFrame::OnMenuConfigView()
{
    // TODO: Add your command handler code here
}
void CMainFrame::OnUpdateMenuConfigView(CCmdUI *pCmdUI)
{
    // TODO: Add your command update UI handler code here
    if(!gIsEditFlag)
    {
        pCmdUI->Enable(FALSE);
        return ;
    }
    pCmdUI->Enable(m_wndClassView.IsCanConfigView());
}
void CMainFrame::OnMenuFind()
{
    m_wndClassView.FindInTree();
}
void CMainFrame::OnUpdateMenuFind(CCmdUI *pCmdUI)
{
    pCmdUI->Enable(TRUE);
}
void CMainFrame::OnUpdateMenuSaveToxml(CCmdUI *pCmdUI)
{
    pCmdUI->Enable(TRUE);
}
//////////////////////
void CMainFrame::OnUpdateMenuSave(CCmdUI *pCmdUI)
{
    CComponentsView *pView = (CComponentsView*)GetActiveView();
    if (pView != NULL)
    {
        if(pView->IsModefied())
        {
            pCmdUI->Enable(TRUE);
            return;
        }
    }
```

```
    pCmdUI->Enable(FALSE);
}
//保存
void CMainFrame::OnMenuSaveYqyb()
{
    // TODO: Add your command handler code here
    if(gpSIMView == NULL)
        return ;
    //保存数据
    gpSIMView->SaveData();
}
void CMainFrame::OnUpdateMenuSaveYqyb(CCmdUI *pCmdUI)
{
    // TODO: Add your command update UI handler code here
    CComponentsView *pView = (CComponentsView*)GetActiveView();
    if (pView != NULL)
    {
        if(pView->IsModefied())
        {
            pCmdUI->Enable(TRUE);
            return;
        }
    }
    pCmdUI->Enable(FALSE);
}
void CMainFrame::OnMenuEquiptWh()
{
    // TODO: Add your command handler code here
    CComponentsView *pView = (CComponentsView*)GetActiveView();
    if (pView != NULL)
    {
        OBJECT_ID CurFxtid = m_wndClassView.m_nCurFxtObjid; //当前分系统
的 id
        pView->OnEquitpWh(0);
    }
}
void CMainFrame::OnMenuRefreshTree()//刷新树
{
    if(m_wndClassView.GetCurFxtName() == "总体")
    {
        m_wndClassView.RefreshTree();
    }
    else
    {
```

```
            m_wndClassView.RefreshTree();
        }
}
void CMainFrame::OnMenuClassTreeFocus()//设置 ClassView 树焦点
{
        m_wndClassView.SetClassTreeFocus();
}
void CMainFrame::OnMenuOutTreeFocus()//刷新树
{
}
void CMainFrame::WriteLog(CString m_sLogFilePath, CString sInfo)
{
        if(m_sLogFilePath == "")
                return ;
        CStdioFile mFile;
        mFile.Open( m_sLogFilePath, CFile::modeWrite | CFile::modeCreate |
CFile::modeNoTruncate);
        mFile.SeekToEnd();
        mFile.WriteString(sInfo);
        mFile.WriteString("\r\n");
        mFile.Close();
}
Components.h
// Components.h : main header file for the SIM application
//
#pragma once
#ifndef __AFXWIN_H__
        #error "include 'stdafx.h' before including this file for PCH"
#endif
#include "resource.h"               // main symbols
// CComponentsApp:
// See Components.cpp for the implementation of this class
//
class    CConstraint : public CObject
{
public:
        //CConstraint(void);
        //virtual ~CConstraint(void);
        CStringArray m_cConstraintArray;
};
//end
class CComponentsApp : public CWinAppEx
{
public:
```

```
	CComponentsApp();
public:
	//CJZBase m_cJZBase;
// Overrides
public:
	virtual BOOL InitInstance();
// Implementation
	UINT	m_nAppLook;
	BOOL	m_bHiColorIcons;
	HANDLE	m_hMutex;

	virtual void PreLoadState();
	virtual void LoadCustomState();
	virtual void SaveCustomState();
	afx_msg void OnAppAbout();
	DECLARE_MESSAGE_MAP()
	virtual int ExitInstance();
	CString GetMaxMacAddress();
	void InitSelectContents();
public:
	CStringArray m_cArrayYQJFL;
	CStringArray m_cArrayZLDJ;
	CStringArray m_cArrayZLDJFL;
	CStringArray m_cArrayZLBZDW;
	CStringArray m_cArrayJLDW;
	CMapStringToOb m_cMapConstraintSelect;
};
extern CComponentsApp theApp;
```

Components.cpp

```
// Components.cpp : Defines the class behaviors for the application.
//
#include "stdafx.h"
#include "afxwinappex.h"
#include "Components.h"
#include "MainFrm.h"
#include "ComponentsDoc.h"
#include "ComponentsView.h"
#include <afxdllx.h>
#include "nb30.h"
#include	<atlbase.h>
#include	<atlconv.h>
#include "NTCore_imp.h"
#include "Userlogin_imp.h"
#ifdef _DEBUG
```

```
#define new DEBUG_NEW
#endif
CString gcMacAddr ="";
BOOL gIsEditFlag;
int gSysVerFlag = 0;
extern "C"    __declspec(dllimport) bool gl_DecryptCont(BSTR bstrOldContent,
BSTR *bstrNewContent,int nFlag) ;
// CComponentsApp
BEGIN_MESSAGE_MAP(CComponentsApp, CWinAppEx)
    ON_COMMAND(ID_APP_ABOUT, &CComponentsApp::OnAppAbout)
    // Standard file based document commands
    ON_COMMAND(ID_FILE_NEW, &CWinAppEx::OnFileNew)
    ON_COMMAND(ID_FILE_OPEN, &CWinAppEx::OnFileOpen)
    // Standard print setup command
    ON_COMMAND(ID_FILE_PRINT_SETUP, &CWinAppEx::OnFilePrintSetup)
END_MESSAGE_MAP()
// CComponentsApp construction
CComponentsApp::CComponentsApp()
{
    m_bHiColorIcons = TRUE;

    // TODO: add construction code here,
    // Place all significant initialization in InitInstance
}
// The one and only CComponentsApp object
CComponentsApp theApp;
// CComponentsApp initialization
BOOL CComponentsApp::InitInstance()
{
    // InitCommonControlsEx() is required on Windows XP if an application
    // manifest specifies use of ComCtl32.dll version 6 or later to enable
    // visual styles.    Otherwise, any window creation will fail.
    INITCOMMONCONTROLSEX InitCtrls;
    InitCtrls.dwSize = sizeof(InitCtrls);
    // Set this to include all the common control classes you want to use
    // in your application.
    InitCtrls.dwICC = ICC_WIN95_CLASSES;
    InitCommonControlsEx(&InitCtrls);

    CoInitialize(NULL);
    CWinAppEx::InitInstance();

    // Initialize OLE libraries
    if (!AfxOleInit())
```

```
{
    CString sErrorInfo;
    sErrorInfo.Format(IDP_OLE_INIT_FAILED);
    return FALSE;
}
gcMacAddr = GetMaxMacAddress();

CString strName = "用户管理";
if (gl_UserLogin(strName) == -1)
    return FALSE;
//初始化约束选取内容
InitSelectContents();
AfxEnableControlContainer();
// Standard initialization
// If you are not using these features and wish to reduce the size
// of your final executable, you should remove from the following
// the specific initialization routines you do not need
// Change the registry key under which our settings are stored
// TODO: You should modify this string to be something appropriate
// such as the name of your company or organization
SetRegistryKey(_T("Local AppWizard-Generated Applications"));
LoadStdProfileSettings(4);  // Load standard INI file options (including MRU)
InitContextMenuManager();
InitKeyboardManager();
InitTooltipManager();
CMFCToolTipInfo ttParams;
ttParams.m_bVislManagerTheme = TRUE;
theApp.GetTooltipManager()->SetTooltipParams(AFX_TOOLTIP_TYPE_ALL,
    RUNTIME_CLASS(CMFCToolTipCtrl), &ttParams);
// Register the application's document templates.  Document templates
//  serve as the connection between documents, frame windows and views
CSingleDocTemplate* pDocTemplate;
pDocTemplate = new CSingleDocTemplate(
    IDR_MAINFRAME,
    RUNTIME_CLASS(CComponentsDoc),
    RUNTIME_CLASS(CMainFrame),          // main SDI frame window
    RUNTIME_CLASS(CComponentsView));
if (!pDocTemplate)
    return FALSE;
AddDocTemplate(pDocTemplate);
// Parse command line for standard shell commands, DDE, file open
CCommandLineInfo cmdInfo;
ParseCommandLine(cmdInfo);
// Dispatch commands specified on the command line.  Will return FALSE if
```

```
	// app was launched with /RegServer, /Register, /Unregserver or /Unregister.
	if (!ProcessShellCommand(cmdInfo))
		return FALSE;
	// The one and only window has been initialized, so show and update it
	m_pMainWnd->ShowWindow(SW_SHOW);
	m_pMainWnd->UpdateWindow();

	// call DragAcceptFiles only if there's a suffix
	//  In an SDI app, this should occur after ProcessShellCommand
	return TRUE;
}
// CAboutDlg dialog used for App About
class CAboutDlg : public CDialog
{
public:
	CAboutDlg();
// Dialog Data
	enum { IDD = IDD_ABOUTBOX };
protected:
	virtual void DoDataExchange(CDataExchange* pDX);	// DDX/DDV support

// Implementation
protected:
	afx_msg void OnLButtonUp(UINT nFlags, CPoint point);
	DECLARE_MESSAGE_MAP()
};
CAboutDlg::CAboutDlg() : CDialog(CAboutDlg::IDD)
{
}
void CAboutDlg::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
}
BEGIN_MESSAGE_MAP(CAboutDlg, CDialog)
	ON_WM_LBUTTONUP()
END_MESSAGE_MAP()
void CAboutDlg::OnLButtonUp(UINT nFlags, CPoint point)
{
	CDialog::OnOK();
}
// App command to run the dialog
void CComponentsApp::OnAppAbout()
{
	CAboutDlg aboutDlg;
```

```
	aboutDlg.DoModal();
}
// CComponentsApp customization load/save methods
void CComponentsApp::PreLoadState()
{
	BOOL bNameValid;
	CString strName;
	bNameValid = strName.LoadString(IDS_EDIT_MENU);
	ASSERT(bNameValid);
	GetContextMenuManager()->AddMenu(strName, IDR_POPUP_EDIT);
	bNameValid = strName.LoadString(IDS_EXPLORER);
	ASSERT(bNameValid);
	GetContextMenuManager()->AddMenu(strName, IDR_POPUP_EXPLORER);
}
void CComponentsApp::LoadCustomState()
{
}
void CComponentsApp::SaveCustomState()
{
}
// CComponentsApp message handlers
int CComponentsApp::ExitInstance()
{
	// TODO: Add your specialized code here and/or call the base clas
	m_cArrayYQJFL.RemoveAll();
	m_cArrayZLDJ.RemoveAll();
	m_cArrayZLDJFL.RemoveAll();
	m_cArrayZLBZDW.RemoveAll();
	m_cArrayJLDW.RemoveAll();

	POSITION posTemp2 = m_cMapConstraintSelect.GetStartPosition();
	while (posTemp2)
	{
		CString strAttrName = "";
		CConstraint *pConstraint = NULL;

	m_cMapConstraintSelect.GetNextAssoc(posTemp2,strAttrName,(CObject*&)pC
onstraint);
		if(pConstraint != NULL)
			delete pConstraint;
	}
	m_cMapConstraintSelect.RemoveAll();
	return CWinAppEx::ExitInstance();
}
```

```
void CComponentsApp::InitSelectContents()
{
	CString sFileNameTemp;
	char path[256];
	GetModuleFileName(AfxGetInstanceHandle(),path,256);
	sFileNameTemp = path;
	int nIndex = sFileNameTemp.ReverseFind('\\');
	sFileNameTemp = sFileNameTemp.Left(nIndex);
	CString sFileYueShuIni;
	sFileYueShuIni.Format("%s\\约束.ini",sFileNameTemp);
	CFileFind cFileFind;
	if(cFileFind.FindFile(sFileYueShuIni))
	{
		cFileFind.Close();

		CStringArray m_sAttrYs;
		char szAttrCount[255] = "\0";
		GetPrivateProfileString("约束属性","数量
","0",szAttrCount,256,sFileYueShuIni);
		int nAttrCount = 0;
		nAttrCount = atoi(szAttrCount);
		if(nAttrCount >0)
		{
			m_sAttrYs.RemoveAll();
			for(int i = 1;i <= nAttrCount;i++)
			{
				char temp[100]="\0";
				sprintf(temp,"属性%d",i);
				char szOutNameValue[100]="\0";
				GetPrivateProfileString("约束属性
",temp,"0",szOutNameValue,256,sFileYueShuIni);
				CString strOutNameValue = szOutNameValue;
				if(strOutNameValue != "")
					m_sAttrYs.Add(_T(strOutNameValue));
			}
			for(int j = 0; j < m_sAttrYs.GetCount(); j++)
			{
				CString strAttrName = m_sAttrYs.GetAt(j);
				char szCount[255] = "\0";
				GetPrivateProfileString(strAttrName,"数量
","0",szCount,256,sFileYueShuIni);
				int nYqjCount = 0;
				nYqjCount = atoi(szCount);
				if(nYqjCount <= 0)
```

```
                continue;
            CConstraint *pConstraint = NULL;

    m_cMapConstraintSelect.Lookup(strAttrName,(CObject*&)pConstraint);
            if (pConstraint == NULL)
            {
                pConstraint = new CConstraint;
                m_cMapConstraintSelect.SetAt(strAttrName,pConstraint);
            }

            CString strYsValue = "";
            for(int i = 1;i <= nYqjCount;i++)
            {
                char temp[100]="\0";
                sprintf(temp,"约束%d",i);
                char szOutValue[100]="\0";

    GetPrivateProfileString(strAttrName,temp,"0",szOutValue,256,sFileYueShuIni);
                CString strOutValue = szOutValue;
                if(strOutValue != "")
                {
                    if(pConstraint != NULL)
                        pConstraint->m_cConstraintArray.Add(strOutValue);
                }
            }
        }
    }

    }
}
ComponentsView.h
// CComponentsView.h : interface of the CSIMView class
//
#pragma once
#include "ComponentsDoc.h"
#include "DlgClsDetail.h"
class CComponentsView : public CView
{
protected: // create from serialization only
    CComponentsView();
    DECLARE_DYNCREATE(CComponentsView)
// Attributes
public:
    CComponentsDoc* GetDocument() const;
```

```
// Operations
public:
// Overrides
public:
    virtual void OnDraw(CDC* pDC);    // overridden to draw this view
    virtual BOOL PreCreateWindow(CREATESTRUCT& cs);
protected:
    virtual BOOL OnPreparePrinting(CPrintInfo* pInfo);
    virtual void OnBeginPrinting(CDC* pDC, CPrintInfo* pInfo);
    virtual void OnEndPrinting(CDC* pDC, CPrintInfo* pInfo);
// Implementation
public:
    virtual ~CComponentsView();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
public:
    BOOL SaveData();
    BOOL IsModefied();//判断修改信息是否保存到 xml
    void RepaintCombox();
    BOOL ShowInfo(OBJECT_ID idObj,OBJECT_ID m_nCurFxtObjid, CString
sQdType,CString strFxtName);
    void OnEquitpWh(OBJECT_ID CurFxtid);   //设备维护
    BOOL ShowZjInfo(OBJECT_ID idObj,OBJECT_ID m_nCurFxtObjid,CString
sQdType,CString strFxtName);
    CString m_strEquiptEdit;   //设备维护中“编辑”按钮是否灰显
    CString m_strEquiptNew;   //设备维护中“新增”按钮是否灰显
    CString m_strEquiptDelete;   //设备维护中“删除”按钮是否灰显
public:
    CDlgClsDetail *m_pDlgClsDetail;
// Generated message map functions
protected:
    afx_msg void OnFilePrintPreview();
    afx_msg void OnRButtonUp(UINT nFlags, CPoint point);
    afx_msg void OnContextMenu(CWnd* pWnd, CPoint point);
    DECLARE_MESSAGE_MAP()
public:
    virtual void OnInitialUpdate();
    afx_msg void OnSize(UINT nType, int cx, int cy);
    afx_msg void OnUpdateProcedureStore(CCmdUI *pCmdUI);
    afx_msg void OnMenuExecSql();
    afx_msg void OnMenuNewYqyb();
    afx_msg void OnUpdateMenuNewYqyb(CCmdUI *pCmdUI);
```

```
    afx_msg void OnMenuDelYqyb();
    afx_msg void OnUpdateMenuDelYqyb(CCmdUI *pCmdUI);
    afx_msg void OnMenuExportYqybExcel();
    afx_msg void OnUpdateMenuExportYqybExcel(CCmdUI *pCmdUI);
    afx_msg void OnMenuImportYqybExcel();
    afx_msg void OnUpdateMenuImportYqybExcel(CCmdUI *pCmdUI);
    afx_msg void OnMenuNew();
    afx_msg void OnUpdateMenuNew(CCmdUI *pCmdUI);
    afx_msg void OnMenuDel();
    afx_msg void OnUpdateMenuDel(CCmdUI *pCmdUI);
    afx_msg void OnMenuExportExcel();
    afx_msg void OnUpdateMenuExportExcel(CCmdUI *pCmdUI);
    afx_msg void OnMenuImportExcel();
    afx_msg void OnUpdateMenuImportExcel(CCmdUI *pCmdUI);
    afx_msg void OnMenuReferFile();
    afx_msg void OnMenuImportEquipt();
    afx_msg void OnUpdateMenuImportEquipt(CCmdUI *pCmdUI);
    afx_msg void OnMenuExportEquipt();
    afx_msg void OnUpdateMenuExportEquipt(CCmdUI *pCmdUI);
    afx_msg void OnMenuImportPl();
    afx_msg void OnMenuXhWh();
    afx_msg void OnUpdateMenuXhWh(CCmdUI *pCmdUI);
    afx_msg void OnMenuExportPl();
    afx_msg void OnUpdateMenuExportPl(CCmdUI *pCmdUI);
    afx_msg void OnMenuSelecttozj();
    afx_msg void OnUpdateMenuSelecttozj(CCmdUI *pCmdUI);
    afx_msg void OnMenuQuery();
    afx_msg void OnMenuYqjbl();
    afx_msg void OnUpdateMenuYqjbl(CCmdUI *pCmdUI);
    afx_msg void OnMenuZlyqj();
    afx_msg void OnUpdateMenuZlyqj(CCmdUI *pCmdUI);
    afx_msg void OnMenuGchbl();
    afx_msg void OnUpdateMenuGchbl(CCmdUI *pCmdUI);
    afx_msg void OnMenuQdCompare();
    afx_msg void OnUpdateMenuQdCompare(CCmdUI *pCmdUI);
    afx_msg void OnMenuZldjtj();
    afx_msg void OnUpdateMenuZldjtj(CCmdUI *pCmdUI);
    afx_msg void OnMenuUserWh();
    afx_msg void OnUpdateMenuUserWh(CCmdUI *pCmdUI);
    afx_msg void OnMenuYqjsum();
    afx_msg void OnMenuQdCompareex();
};
#ifndef _DEBUG  // debug version in SIMView.cpp
inline CComponentsDoc* CComponentsView::GetDocument() const
```

```
    { return reinterpret_cast<CComponentsDoc*>(m_pDocument); }
#endif
```

ComponentsView.cpp

```
// CComponentsView.cpp : implementation of the CComponentsView class
//
#include "stdafx.h"
#include "Components.h"
#include "ComponentsDoc.h"
#include "ComponentsView.h"
#include "DlgReferFile.h"
#include "MainFrm.h"
#include "CAPPGlobalEnv.h"
#include "DlgXhWh.h"
#include "DlgQueryConditions.h"
#include "DlgSelectEquipt.h"
#include "DlgUserWh.h"
#include "PlExportMdb.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
// CComponentsView
_declspec(dllimport) CCAPPGlobalEnv gcCAPPGlobalEnv;
extern BOOL gIsEditFlag;
CComponentsView *gpSIMView = NULL;
IMPLEMENT_DYNCREATE(CComponentsView, CView)
BEGIN_MESSAGE_MAP(CComponentsView, CView)
    // Standard printing commands
    ON_COMMAND(ID_FILE_PRINT, &CView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_DIRECT, &CView::OnFilePrint)
    ON_COMMAND(ID_FILE_PRINT_PREVIEW,
&CComponentsView::OnFilePrintPreview)
    ON_WM_SIZE()
ON_COMMAND(ID_MENU_NEW_YQYB,
&CComponentsView::OnMenuNewYqyb)
ON_UPDATE_COMMAND_UI(ID_MENU_NEW_YQYB,
&CComponentsView::OnUpdateMenuNewYqyb)
ON_COMMAND(ID_MENU_DEL_YQYB,
&CComponentsView::OnMenuDelYqyb)
ON_UPDATE_COMMAND_UI(ID_MENU_DEL_YQYB,
&CComponentsView::OnUpdateMenuDelYqyb)
ON_COMMAND(ID_MENU_EXPORT_YQYB_EXCEL,
&CComponentsView::OnMenuExportYqybExcel)
ON_UPDATE_COMMAND_UI(ID_MENU_EXPORT_YQYB_EXCEL,
&CComponentsView::OnUpdateMenuExportYqybExcel)
```

```
ON_COMMAND(ID_MENU_IMPORT_YQYB_EXCEL,
&CComponentsView::OnMenuImportYqybExcel)
ON_UPDATE_COMMAND_UI(ID_MENU_IMPORT_YQYB_EXCEL,
&CComponentsView::OnUpdateMenuImportYqybExcel)
ON_COMMAND(ID_MENU_NEW, &CComponentsView::OnMenuNew)
ON_UPDATE_COMMAND_UI(ID_MENU_NEW,
&CComponentsView::OnUpdateMenuNew)
ON_COMMAND(ID_MENU_DEL, &CComponentsView::OnMenuDel)
ON_UPDATE_COMMAND_UI(ID_MENU_DEL,
&CComponentsView::OnUpdateMenuDel)
ON_COMMAND(ID_MENU_EXPORT_EXCEL,
&CComponentsView::OnMenuExportExcel)
ON_UPDATE_COMMAND_UI(ID_MENU_EXPORT_EXCEL,
&CComponentsView::OnUpdateMenuExportExcel)
ON_COMMAND(ID_MENU_IMPORT_EXCEL,
&CComponentsView::OnMenuImportExcel)
ON_UPDATE_COMMAND_UI(ID_MENU_IMPORT_EXCEL,
&CComponentsView::OnUpdateMenuImportExcel)
ON_COMMAND(ID_MENU_REFER_FILE,
&CComponentsView::OnMenuReferFile)
ON_COMMAND(ID_MENU_IMPORT_EQUIPT,
&CComponentsView::OnMenuImportEquipt)
ON_UPDATE_COMMAND_UI(ID_MENU_IMPORT_EQUIPT,
&CComponentsView::OnUpdateMenuImportEquipt)
ON_COMMAND(ID_MENU_EXPORT_EQUIPT,
&CComponentsView::OnMenuExportEquipt)
ON_UPDATE_COMMAND_UI(ID_MENU_EXPORT_EQUIPT,
&CComponentsView::OnUpdateMenuExportEquipt)
ON_COMMAND(ID_MENU_IMPORT_PL,
&CComponentsView::OnMenuImportPl)
ON_COMMAND(ID_MENU_XH_WH, &CComponentsView::OnMenuXhWh)
ON_UPDATE_COMMAND_UI(ID_MENU_XH_WH,
&CComponentsView::OnUpdateMenuXhWh)
ON_COMMAND(ID_MENU_EXPORT_PL,
&CComponentsView::OnMenuExportPl)
ON_UPDATE_COMMAND_UI(ID_MENU_EXPORT_PL,
&CComponentsView::OnUpdateMenuExportPl)
ON_COMMAND(ID_MENU_SELECTTOZJ,
&CComponentsView::OnMenuSelecttozj)
ON_UPDATE_COMMAND_UI(ID_MENU_SELECTTOZJ,
&CComponentsView::OnUpdateMenuSelecttozj)
ON_COMMAND(ID_MENU_QUERY, &CComponentsView::OnMenuQuery)
ON_COMMAND(ID_MENU_YQJBL, &CComponentsView::OnMenuYqjbl)
ON_UPDATE_COMMAND_UI(ID_MENU_YQJBL,
```

```
&CComponentsView::OnUpdateMenuYqjbl)
ON_COMMAND(ID_MENU_ZLYQJ, &CComponentsView::OnMenuZlyqj)
ON_UPDATE_COMMAND_UI(ID_MENU_ZLYQJ,
&CComponentsView::OnUpdateMenuZlyqj)
ON_COMMAND(ID_MENU_GCHBL, &CComponentsView::OnMenuGchbl)
ON_UPDATE_COMMAND_UI(ID_MENU_GCHBL,
&CComponentsView::OnUpdateMenuGchbl)
ON_COMMAND(ID_MENU_QD_COMPARE,
&CComponentsView::OnMenuQdCompare)
ON_UPDATE_COMMAND_UI(ID_MENU_QD_COMPARE,
&CComponentsView::OnUpdateMenuQdCompare)
ON_COMMAND(ID_MENU_ZLDJTJ, &CComponentsView::OnMenuZldjtj)
ON_UPDATE_COMMAND_UI(ID_MENU_ZLDJTJ,
&CComponentsView::OnUpdateMenuZldjtj)
ON_COMMAND(ID_MENU_USER_WH, &CComponentsView::OnMenuUserWh)
ON_UPDATE_COMMAND_UI(ID_MENU_USER_WH,
&CComponentsView::OnUpdateMenuUserWh)
ON_COMMAND(ID_MENU_YQJSUM, &CComponentsView::OnMenuYqjsum)
ON_COMMAND(ID_MENU_QD_COMPAREEX,
&CComponentsView::OnMenuQdCompareex)
END_MESSAGE_MAP()
// CComponentsView construction/destruction
CComponentsView::CComponentsView()
{
    // TODO: add construction code here
    m_pDlgClsDetail = NULL;
    m_strEquiptEdit = "是";
    m_strEquiptNew = "是";
    m_strEquiptDelete = "是";
}
CComponentsView::~CComponentsView()
{

}
BOOL CComponentsView::PreCreateWindow(CREATESTRUCT& cs)
{
    // TODO: Modify the Window class or styles here by modifying
    //  the CREATESTRUCT cs
    return CView::PreCreateWindow(cs);
}
// CComponentsView drawing
void CComponentsView::OnDraw(CDC* /*pDC*/)
{
    CComponentsDoc* pDoc = GetDocument();
```

```
    ASSERT_VALID(pDoc);
    if (!pDoc)
        return;
    // TODO: add draw code for native data here
}
// CComponentsView printing
void CComponentsView::OnFilePrintPreview()
{
    AFXPrintPreview(this);
}
BOOL CComponentsView::OnPreparePrinting(CPrintInfo* pInfo)
{
    // default preparation
    return DoPreparePrinting(pInfo);
}
void CComponentsView::OnBeginPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
    // TODO: add extra initialization before printing
}
void CComponentsView::OnEndPrinting(CDC* /*pDC*/, CPrintInfo* /*pInfo*/)
{
    // TODO: add cleanup after printing
}
void CComponentsView::OnRButtonUp(UINT nFlags, CPoint point)
{
    ClientToScreen(&point);
    OnContextMenu(this, point);
}
void CComponentsView::OnContextMenu(CWnd* pWnd, CPoint point)
{
    theApp.GetContextMenuManager()->ShowPopupMenu(IDR_POPUP_EDIT,
point.x, point.y, this, TRUE);
}
// CComponentsView diagnostics
#ifdef _DEBUG
void CComponentsView::AssertValid() const
{
    CView::AssertValid();
}
void CComponentsView::Dump(CDumpContext& dc) const
{
    CView::Dump(dc);
}
CComponentsDoc* CComponentsView::GetDocument() const // non-debug version is
```

```
inline
{
    ASSERT(m_pDocument->IsKindOf(RUNTIME_CLASS(CComponentsDoc)));
    return (CComponentsDoc*)m_pDocument;
}
#endif //_DEBUG
// CComponentsView message handlers
void CComponentsView::OnInitialUpdate()
{
    CView::OnInitialUpdate();
    // TODO: Add your specialized code here and/or call the base class
    m_pDlgClsDetail = new CDlgClsDetail(this);
    m_pDlgClsDetail->Create(this);
    m_pDlgClsDetail->ShowWindow(SW_SHOW);
    CRect rc;
    GetClientRect(rc);
    m_pDlgClsDetail->MoveWindow(rc);
     CString sFileNameTemp;
    char path[256];
    GetModuleFileName(AfxGetInstanceHandle(),path,256);
    sFileNameTemp = path;
    int nIndex = sFileNameTemp.ReverseFind('\\');
    sFileNameTemp = sFileNameTemp.Left(nIndex);
    CString sFileEquiptIni;
    sFileEquiptIni.Format("%s\\Equipt.ini",sFileNameTemp);
    CFileFind cFileFind;
    if(cFileFind.FindFile(sFileEquiptIni))
    {
        cFileFind.Close();
        char szEquiptEdit[255] = "\0";
        char szEquiptNew[255] = "\0";
        char szEquiptDelete[255] = "\0";
        GetPrivateProfileString("设备配套维护","编辑
","0",szEquiptEdit,256,sFileEquiptIni);
        GetPrivateProfileString("设备配套维护","新增
","0",szEquiptNew,256,sFileEquiptIni);
        GetPrivateProfileString("设备配套维护","删除
","0",szEquiptDelete,256,sFileEquiptIni);
        m_strEquiptEdit = szEquiptEdit;
        m_strEquiptNew = szEquiptNew;
        m_strEquiptDelete = szEquiptDelete;
    }
    gpSIMView = this;
}
```

```
void CComponentsView::OnSize(UINT nType, int cx, int cy)
{
    CView::OnSize(nType, cx, cy);
    // TODO: Add your message handler code here
    if (m_pDlgClsDetail != NULL)
    {
        CRect rc;
        GetClientRect(rc);
        m_pDlgClsDetail->MoveWindow(rc);
    }
}
void CComponentsView::RepaintCombox()
{
}
//表格是否修改
BOOL CComponentsView::IsModefied()
{
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 )
    {
        CDlgPtInst *pDlgPtInst = m_pDlgClsDetail->GetPtInstDlg();
        ASSERT(pDlgPtInst != NULL);
        if(pDlgPtInst->IsModified())
        {
            return TRUE;
        }
    }
    else if(nCurSelect == 1 )
    {
        CDlgEquiptModify *pDlgEquiptInst =
m_pDlgClsDetail->GetEquitpInstDlg();
        if(pDlgEquiptInst == NULL)
            return FALSE;
        if(pDlgEquiptInst->IsModified())
        {
            return TRUE;
        }
    }
    return FALSE;
}
//刷新元器件表格
//OBJECT_ID idObj 当前所选树节点的 id；OBJECT_ID m_nCurFxtObjid 当前分
系统的 id;
BOOL CComponentsView::ShowInfo(OBJECT_ID idObj,OBJECT_ID
```

```
m_nCurFxtObjid,CString sQdType,CString strFxtName)
{
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 2 || nCurSelect == 1 )
    {
        m_pDlgClsDetail->SelectTab(0,TRUE);
    }
    CDlgPtInst *pDlgPtInst = m_pDlgClsDetail->GetPtInstDlg();
    ASSERT(pDlgPtInst != NULL);
    pDlgPtInst->RepaintInst(idObj,m_nCurFxtObjid,strFxtName);

    return TRUE;
}
//刷新元器件表格
//OBJECT_ID idObj  当前所选树节点的 id；OBJECT_ID m_nCurFxtObjid  当前分
系统的 id; sQdType 清单类型，为选用清单，为装机清单
BOOL CComponentsView::ShowZjInfo(OBJECT_ID idObj,OBJECT_ID
m_nCurFxtObjid,CString sQdType,CString strFxtName)
{
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 1 || nCurSelect == 0)
    {
        m_pDlgClsDetail->SelectTab(1,TRUE);
    }
    return TRUE;
}
//保存
BOOL CComponentsView::SaveData()
{
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 )
    {
        CDlgPtInst *pDlgPtInst = m_pDlgClsDetail->GetPtInstDlg();
        ASSERT(pDlgPtInst != NULL);

        BOOL bSaveFlag = pDlgPtInst->SaveData(TRUE);
        return bSaveFlag;
    }
    else if(nCurSelect == 1 )
    {
        CDlgEquiptModify *pDlgEquiptInst =
m_pDlgClsDetail->GetEquitpInstDlg();
        if(pDlgEquiptInst == NULL)
            return FALSE;
```

```
            pDlgEquiptInst->SaveData(TRUE);
            CMainFrame *pMain = (CMainFrame*)AfxGetMainWnd();
            if(pMain == NULL)
                return FALSE;
            pMain->OnMenuRefreshTree();
        }
        return TRUE;
}
//新增
void CComponentsView::OnMenuNew()
{
        // TODO: Add your command handler code here
        int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
        if(nCurSelect == 0 )
        {
            CDlgPtInst *pDlgPtInst = m_pDlgClsDetail->GetPtInstDlg();
            if(pDlgPtInst == NULL)
                return ;
            pDlgPtInst->NewInst();
        }
        else if(nCurSelect == 1 )
        {
            CDlgEquiptModify *pDlgEquiptInst =
m_pDlgClsDetail->GetEquitpInstDlg();
            if(pDlgEquiptInst == NULL)
                return ;
            pDlgEquiptInst->NewInst();
        }
}
void CComponentsView::OnUpdateMenuNew(CCmdUI *pCmdUI)
{
        // TODO: Add your command update UI handler code here
        CString strUserName = gcCAPPGlobalEnv.GetCurrentUser()->GetUserAlias();
//用户名，相当于分系统
        int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
        if(nCurSelect == 0)
        {
            CDlgPtInst *pDlgPtInst = m_pDlgClsDetail->GetPtInstDlg();
            if(pDlgPtInst == NULL)
                return ;
            CString sFxtType = pDlgPtInst->GetFxtType();
            if(sFxtType == "1")   //分系统下，不能新增。
                pCmdUI->Enable(FALSE);
            else
```

```
                pCmdUI->Enable(TRUE);
            return ;
        }
        else if(nCurSelect == 1 )
        {
            CDlgEquiptModify *pDlgEquiptInst =
m_pDlgClsDetail->GetEquitpInstDlg();
            if(pDlgEquiptInst == NULL)
                return ;
            BOOL bEdit = pDlgEquiptInst->GetIsEdit();
            if(strUserName == "总体")
            {
                if (!bEdit)
                {
                    pCmdUI->Enable(FALSE);
                }
                else
                {
                    pCmdUI->Enable(TRUE);
                }
            }
            else
            {
                if(m_strEquiptNew == "是")
                {
                    if (!bEdit)
                    {
                        pCmdUI->Enable(FALSE);
                    }
                    else
                    {
                        pCmdUI->Enable(TRUE);
                    }
                }
                else
                {
                    pCmdUI->Enable(FALSE);
                    return ;
                }
            }
            return ;
        }
        else
        {
```

```
            pCmdUI->Enable(FALSE);
        }
}
//删除
void CComponentsView::OnMenuDel()
{
    // TODO: Add your command handler code here
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 )
    {
        CDlgPtInst *pDlgPtInst = m_pDlgClsDetail->GetPtInstDlg();
        ASSERT(pDlgPtInst != NULL);
        pDlgPtInst->DelInst();
    }

    else if(nCurSelect == 1 )
    {
        CDlgEquiptModify *pDlgEquiptInst =
m_pDlgClsDetail->GetEquitpInstDlg();
        if(pDlgEquiptInst == NULL)
            return ;
        pDlgEquiptInst->DelInst();
        CMainFrame *pMain = (CMainFrame*)AfxGetMainWnd();
        if(pMain == NULL)
            return ;
        pMain->OnMenuRefreshTree();
    }
}
void CComponentsView::OnUpdateMenuDel(CCmdUI *pCmdUI)
{
    // TODO: Add your command update UI handler code here
    CString strUserName = gcCAPPGlobalEnv.GetCurrentUser()->GetUserAlias();
//用户名，相当于分系统
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 )
    {
        CDlgPtInst *pDlgPtInst = m_pDlgClsDetail->GetPtInstDlg();
        ASSERT(pDlgPtInst != NULL);
        CString sFxtType = pDlgPtInst->GetFxtType();
        if(sFxtType == "3")
        {
            pCmdUI->Enable(FALSE);
        }
        int nRow = pDlgPtInst->GetCurRow();
```

```
            if (nRow <= 0)
            {
                pCmdUI->Enable(FALSE);
                return ;
            }
        }
        else if(nCurSelect == 1 )
        {
            CDlgEquiptModify *pDlgEquiptInst =
m_pDlgClsDetail->GetEquitpInstDlg();
            if(pDlgEquiptInst == NULL)
                return ;
            if(strUserName == "总体")
            {

                int nRow = pDlgEquiptInst->GetCurRow();
                if (nRow <= 0)
                {
                    pCmdUI->Enable(FALSE);
                    return ;
                }
            }
            else
            {
                if(m_strEquiptDelete == "是")
                {
                    int nRow = pDlgEquiptInst->GetCurRow();
                    if (nRow <= 0)
                    {
                        pCmdUI->Enable(FALSE);
                        return ;
                    }
                }
                else
                {
                    pCmdUI->Enable(FALSE);
                    return ;
                }
            }
        }
        else
        {
            pCmdUI->Enable(FALSE);
        }
```

```
}
extern int gSysVerFlag;
//参考文件
void CComponentsView::OnMenuReferFile()
{
    // TODO: Add your command handler code here
    CDlgReferFile    dlgRefFile;
    if (dlgRefFile.DoModal() != IDOK)
        return ;
}
//设备维护。CurFxtid 分系统的 id。 CurFxtid 值为，则列型号下的配套。不为，则列当前分系统的配套
void CComponentsView::OnEquitpWh(OBJECT_ID CurFxtid)
{
    CString strUserName = gcCAPPGlobalEnv.GetCurrentUser()->GetUserAlias();
//用户名，相当于分系统
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 )
    {
        m_pDlgClsDetail->SelectTab(1,TRUE);
    }
    else if(nCurSelect == 1 )
    {
        m_pDlgClsDetail->SelectTab(1,TRUE);
    }
    else if(nCurSelect == 2 )
    {
        m_pDlgClsDetail->SelectTab(1,TRUE);
    }
    CDlgEquiptModify *pDlgEquiptInst = m_pDlgClsDetail->GetEquitpInstDlg();
    if(pDlgEquiptInst == NULL)
        return ;
    BOOL bEdit = TRUE ;
    if(strUserName == "总体")
        bEdit = TRUE;
    else
    {
        if(m_strEquiptEdit == "是")
            bEdit = TRUE;
        else
            bEdit = FALSE;
    }
}
void CComponentsView::OnUpdateMenuExportEquipt(CCmdUI *pCmdUI)
```

```
{
    // TODO: Add your command update UI handler code here
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 || nCurSelect == 1 || nCurSelect == 3)
    {
        pCmdUI->Enable(FALSE);
    }
    else
    {
        pCmdUI->Enable(TRUE);
    }

}
//型号维护
void CComponentsView::OnMenuXhWh()
{
    // TODO: Add your command handler code here
    CDlgXhWh dlgXh;
    if (dlgXh.DoModal() != IDOK)
        return ;
}
void CComponentsView::OnUpdateMenuXhWh(CCmdUI *pCmdUI)
{
    // TODO: Add your command update UI handler code here
    CString strUserName = gcCAPPGlobalEnv.GetCurrentUser()->GetUserAlias();
//用户名，相当于分系统
    if(strUserName == "总体")
        pCmdUI->Enable(TRUE);
    else
        pCmdUI->Enable(FALSE);

}
//查询
void CComponentsView::OnMenuQuery()
{
    // TODO: Add your command handler code here
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 || nCurSelect == 1 || nCurSelect == 2)
    {
        m_pDlgClsDetail->SelectTab(3,TRUE);
    }
    CDlgQuery    *m_pDlgQuery = m_pDlgClsDetail->GetQueryDlg();
    if(m_pDlgQuery == NULL)
        return ;
```

```
	// 条件对话框
	CDlgQueryConditions	cDlgQueryCondition;
	if (cDlgQueryCondition.DoModal() != IDOK)
		return ;
	int nType = cDlgQueryCondition.m_nQdType;
	CString strWhere = cDlgQueryCondition.m_sWhere;
	CStringArray strEquiptArray;
	m_pDlgQuery->RepaintGrid(nType,strWhere,nType,&strEquiptArray,"","",NUL
L);
}
//元器件比例
void CComponentsView::OnMenuYqjbl()
{
	int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
	if(nCurSelect == 0 || nCurSelect == 1 || nCurSelect == 2)
	{
		m_pDlgClsDetail->SelectTab(3,TRUE);
	}
	CDlgQuery	*m_pDlgQuery = m_pDlgClsDetail->GetQueryDlg();
	if(m_pDlgQuery == NULL)
		return ;
	CDlgSelectEquipt	cDlgSelectEquipt;
	if (cDlgSelectEquipt.DoModal() != IDOK)
		return ;
	int nType = cDlgSelectEquipt.m_nQdType;
	m_pDlgQuery->RepaintGrid(2,"",nType,&cDlgSelectEquipt.m_sSelectEquiptArr
ay,cDlgSelectEquipt.m_strXyqdSql,"",&cDlgSelectEquipt.m_cMapAttrAiasName);
}
void CComponentsView::OnUpdateMenuYqjbl(CCmdUI *pCmdUI)
{
}
//装机元器件统计表
void CComponentsView::OnMenuZlyqj()
{
	int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
	if(nCurSelect == 0 || nCurSelect == 1 || nCurSelect == 2)
	{
		m_pDlgClsDetail->SelectTab(3,TRUE);
	}
	CDlgQuery	*m_pDlgQuery = m_pDlgClsDetail->GetQueryDlg();
	if(m_pDlgQuery == NULL)
		return ;

	CDlgSelectEquipt	cDlgSelectEquipt;
```

```
    if (cDlgSelectEquipt.DoModal() != IDOK)
        return ;
    int nType = cDlgSelectEquipt.m_nQdType;
    m_pDlgQuery->RepaintGrid(3,"",nType,&cDlgSelectEquipt.m_sSelectEquiptArr
ay,cDlgSelectEquipt.m_strXyqdSql,"",&cDlgSelectEquipt.m_cMapAttrAiasName);
}
void CComponentsView::OnUpdateMenuZlyqj(CCmdUI *pCmdUI)
{
    // TODO: Add your command update UI handler code here
}
//国产化比例
void CComponentsView::OnMenuGchbl()
{
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 || nCurSelect == 1 || nCurSelect == 2)
    {
        m_pDlgClsDetail->SelectTab(3,TRUE);
    }
    CDlgQuery    *m_pDlgQuery = m_pDlgClsDetail->GetQueryDlg();
    if(m_pDlgQuery == NULL)
        return ;

    CDlgSelectEquipt     cDlgSelectEquipt;
    if (cDlgSelectEquipt.DoModal() != IDOK)
        return ;
    int nType = cDlgSelectEquipt.m_nQdType;
    m_pDlgQuery->RepaintGrid(4,"",nType,&cDlgSelectEquipt.m_sSelectEquiptArr
ay,cDlgSelectEquipt.m_strXyqdSql,"",&cDlgSelectEquipt.m_cMapAttrAiasName);
}
void CComponentsView::OnUpdateMenuGchbl(CCmdUI *pCmdUI)
{
}
void CComponentsView::OnMenuZldjtj()
{
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 || nCurSelect == 1 || nCurSelect == 2)
    {
        m_pDlgClsDetail->SelectTab(3,TRUE);
    }
    CDlgQuery    *m_pDlgQuery = m_pDlgClsDetail->GetQueryDlg();
    if(m_pDlgQuery == NULL)
        return ;

    CDlgSelectEquipt     cDlgSelectEquipt;
```

```
    if (cDlgSelectEquipt.DoModal() != IDOK)
        return ;
    int nType = cDlgSelectEquipt.m_nQdType;
    m_pDlgQuery->RepaintGrid(6,"",nType,&cDlgSelectEquipt.m_sSelectEquiptArr
ay,cDlgSelectEquipt.m_strXyqdSql,"",&cDlgSelectEquipt.m_cMapAttrAiasName);
}
void CComponentsView::OnUpdateMenuZldjtj(CCmdUI *pCmdUI)
{
}
//用户维护
void CComponentsView::OnMenuUserWh()
{
    // TODO: Add your command handler code here
    CDlgUserWh dlgUserwh;
    dlgUserwh.DoModal();

    CMainFrame *pMain = (CMainFrame*)AfxGetMainWnd();
    if(pMain == NULL)
        return ;
    pMain->OnMenuRefreshTree();
}
//用户维护
void CComponentsView::OnUpdateMenuUserWh(CCmdUI *pCmdUI)
{
    // TODO: Add your command update UI handler code here
    CString strUserName = gcCAPPGlobalEnv.GetCurrentUser()->GetUserAlias();
//用户名，相当于分系统
    if(strUserName == "总体")
        pCmdUI->Enable(TRUE);
    else
        pCmdUI->Enable(FALSE);
}
void CComponentsView::OnMenuYqjsum()
{
    // TODO: Add your command handler code here
    int nCurSelect = m_pDlgClsDetail->GetCurSelectTab();
    if(nCurSelect == 0 || nCurSelect == 1 || nCurSelect == 2)
    {
        m_pDlgClsDetail->SelectTab(3,TRUE);
    }
    CDlgQuery    *m_pDlgQuery = m_pDlgClsDetail->GetQueryDlg();
    if(m_pDlgQuery == NULL)
        return ;
    CDlgSelectEquipt     cDlgSelectEquipt;
```

```
    if (cDlgSelectEquipt.DoModal() != IDOK)
        return ;
    int nType = cDlgSelectEquipt.m_nQdType;
    m_pDlgQuery->RepaintGrid(7,"",nType,&cDlgSelectEquipt.m_sSelectEquiptArr
ay,cDlgSelectEquipt.m_strXyqdSql,"",&cDlgSelectEquipt.m_cMapAttrAiasName);
}
```

ComponentsDoc.h

```
// CComponentsDoc.h : interface of the CSIMDoc class
//
#pragma once
class CComponentsDoc : public CDocument
{
protected: // create from serialization only
    CComponentsDoc();
    DECLARE_DYNCREATE(CComponentsDoc)
// Attributes
public:
// Operations
public:
// Overrides
public:
    virtual BOOL OnNewDocument();
    virtual void Serialize(CArchive& ar);
// Implementation
public:
    virtual ~CComponentsDoc();
#ifdef _DEBUG
    virtual void AssertValid() const;
    virtual void Dump(CDumpContext& dc) const;
#endif
protected:
// Generated message map functions
protected:
    DECLARE_MESSAGE_MAP()
};
```

ComponentsDoc.cpp

```
// CComponentsDoc.cpp : implementation of the CSIMDoc class
//
#include "stdafx.h"
#include "Components.h"
#include "ComponentsDoc.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#endif
```

```
// CComponentsDoc
IMPLEMENT_DYNCREATE(CComponentsDoc, CDocument)
BEGIN_MESSAGE_MAP(CComponentsDoc, CDocument)
END_MESSAGE_MAP()
// CComponentsDoc construction/destruction
CComponentsDoc::CComponentsDoc()
{
    // TODO: add one-time construction code here
}
CComponentsDoc::~CComponentsDoc()
{
}
BOOL CComponentsDoc::OnNewDocument()
{
    if (!CDocument::OnNewDocument())
        return FALSE;
    // TODO: add reinitialization code here
    // (SDI documents will reuse this document)
    return TRUE;
}
// CComponentsDoc serialization
void CComponentsDoc::Serialize(CArchive& ar)
{
    if (ar.IsStoring())
    {
        // TODO: add storing code here
    }
    else
    {
        // TODO: add loading code here
    }
}
// CComponentsDoc diagnostics
#ifdef _DEBUG
void CComponentsDoc::AssertValid() const
{
    CDocument::AssertValid();
}
void CComponentsDoc::Dump(CDumpContext& dc) const
{
    CDocument::Dump(dc);
}
#endif //_DEBUG
// CComponentsDoc commands
```