石家庄铁道大学毕业设计

合同管理系统

Contract Manage System

2015 届 信息科学与技术 学院

专 业 网络工程

学 号 00000000000

学生姓名 ###

指导老师 ##

完成日期 2015 年 月 日

毕业设计成绩单

学生姓名	***	学号	12312	班级		专业	
毕业设计题目	合同管理系统						
指导教师姓名							
指导教师职称	讲师						
评 定 成 绩							
指导教师				得分			
评阅人				得分			
答辩小组组长				得分			

成绩：

院长(主任) 签字：

年 月 日

毕业设计任务书

<table>
<tr><td>题 目</td><td colspan="7">合同管理系统</td></tr>
<tr><td>学生姓名</td><td></td><td>学号</td><td></td><td>班级</td><td></td><td>专业</td><td></td></tr>
<tr><td colspan="2">承担指导任务单位</td><td colspan="2">信息学院</td><td>导师姓名</td><td></td><td>导师职称</td><td>讲师</td></tr>
<tr><td colspan="8">一、主要内容：
系统主要分为两个大的模块：系统设置模块和合同管理模块。
合同管理模块:单位设置，对签约单位基本属性的设置，遍历起草合同时增加的来往单位基本信息，并对其进行修改保存；增加合同，起草合同的首页基本属性，资金计划，合同执行情况，合同文本，合同执行来往单位及备注；修改合同，对已经起草的合同进行修改，从目录遍历已经起草完成的合同对合同状态进行逐级提交；合同类别设置，增删改查合同类别，可导入导出 EXCEL 文件
系统设置模块：权限设置，增加删除操作人员并设置操作人员的操作权限；修改密码，修改当前操作员登录密码及基本信息；端口设置，网络连接设置，端口、ip 地址的修改；设置时间，系统当前时间设置；升级，导入系统升级文件。
二、基本要求：
1.适用于公司签署的所有经济类合同的管理；
2.规范合同的签订、履行和付款程序，并最大限度的避免风险；
3.友好的界面，方便用户操作；免安装对硬件要求低(32 位 xp 系统及更高版本操作系统)，方便移植；
三、主要技术指标
(1).学习 Delphi 语言，并熟练掌握 Delphi2009 环境配置及软件使用。
(2).熟悉合同签订审批流程及合同管理的相关知识，分析系统需求；
(3).构思系统整体框架，补充功能模块，实现系统功能，美化界面，达到软件用户体验良好的效果。
四、应收集的资料及参考文献
[1].张岭,宋坤,梁冰.Delphi 程序开发典例宝典,2006.7.
[2].罗斌.Delphi for.NET 编程实例精粹,2006.5.
[3].明日科技,梁冰,宋坤.Delphi 范例完全自学手册.
五、进度计划
第 1 周——第 6 周:毕业实习，开题，查询相关资料。
第 7 周——第 8 周: 应用分析、应用设计(开发环境、开发工具的确定熟悉开发环境及工具、功能模块设计、代码设计)。
第 9 周——第 13 周: 程序开发(编程及调试)。
第 14 周——第 16 周: 论文书写及答辩。</td></tr>
<tr><td colspan="2">教研室主任签字</td><td></td><td colspan="2">时 间</td><td colspan="3">年 月 日</td></tr>
</table>

毕业设计开题报告

题 目	合同管理系统						
学生姓名		学号		班级		专业	

一、课题的研究背景及意义

采用手工管理合同，由于涉及的部门众多，需要管理的合同要素也各不相同，因此造成信息不集中，实时性不强，导致各部门协作，业务流程组建，监控制度执行方面效率不高，费时费力等问题，具体表现在如下方面：

1. 文档管理困难：传统纸质合同与电子版合同共存，但对于不同的人员想阅读参考合同时，存在查找不方便的问题。尤其是领导需要了解合同文本时需要耗费很多时间。

2. 进度控制困难：由于合同数目多，参与人员多，合同进度的控制基本靠手工和普通 word、excel 管理已很难满足公司发展需要，并且当领导想全局或全程了解合同情况时存在很大障碍。财务人员的付款依据也与进度密切相关，但同样存在障碍。

3. 信息汇总困难：采用手工或 EXCEL 管理时，由于不同部门的数据格式不统一，采集也不能够及时继续，汇总工作需要耗费大量时间还不一定准确。对于领导的决策时间有一定的影响。

4. 缺少预警机制：缺少对合同进度、结款等关键节点的预警，不能准确地预测近期可能的收支项目，不能帮助公司进行财务规划，掌控现金流，更好地发挥资金运作。

二、国内外研究现状

1. 规范基础数据管理

合同管理系统采用集中的数据管理，可以有效地改变企业信息分布比较散乱的现状。同时，系统提供自定义字段功能，可以为企业实现完整、规范的数据管理提供一个良好的平台支撑。

2. 提高管理效率

合同管理系统避免了手工管理合同出现，利用审批流管理，可以方便、快捷地处理企业管理事务；预警平台可以协助合同管理人员更轻松地应对日常管理事务；报表管理的灵活定义，为管理部门的统计分析提供强大的技术支持。

3. 实现标准化管理和个性化管理的有机结合

合同管理系统通过全局设置和权限分配，可以由公司制订统一采用的管理标准，比如，由公司设置业务单据必填字段，信息录入必须填写这些信息；数据字典可以为某些信息项提供标准的选择项，防止录入人员的录入随意性；设置统一的报表格式，信息输出时可以保证一致性。同时，系统也允许用户根据自身的实际情况采取个性化的措施，比如网格的列显示、自定义报

表等。

4.为领导决策提供准确及时广泛的信息

利用互联网，领导可以在任何时间、任何地方进入系统，随时查阅与合同管理相关基本信息，准确定位下属人员的工作情况，掌握合同执行现状，并方便地生成各种数据报表或图表。同时，系统还提供强大的管理工具模块供客户选择，对这些信息进行深入分析，为公司决策人员提供更多的智力支持。

5.为用户提供全面合同管理解决方案

合同管理系统与企业管理咨询相结合，可以为客户提供全面的合同管理解决方案。比如，通过系统的开放性和灵活性，可以有效地固化专业咨询的工作成果，这就为客户合同管理能力及水平的提升提供足够的平台支持。

总之，通过合同管理系统的应用和实施，可以为合同管理信息化和合同管理能力提升提供一个最佳的结合点，切实打造企业核心竞争力。

三、主要工作和所用方法

合同管理模块:单位设置，对签约单位基本属性的设置，遍历起草合同时增加的来往单位基本信息，并对其进行修改保存；增加合同，起草合同的首页基本属性，资金计划，合同执行情况，合同文本，合同执行来往单位及备注；修改合同，对已经起草的合同进行修改，从目录遍历已经起草完成的合同对合同状态进行逐级提交;合同类别设置,增删改查合同类别,可导入导出 EXCEL 文件。

系统设置模块：权限设置，增加删除操作人员并设置操作人员的操作权限；修改密码，修改当前操作员登录密码及基本信息；端口设置，网络连接设置，端口、ip 地址的修改；设置时间，系统当前时间设置；升级，导入系统升级文件。

四、研究的预期结果

1.技术的可行性

在 Windows 环境下搭建 Delphi 开发平台，整个系统用 Delphi 语言设计开发；数据保存采用数据-结构体-内存流-本地文件的方式保存；数据调用采用本地文件-内存流-调入记录-表格的方式调用已经保存的数据。网络连接采用

2.操作可行性

可应用于各类公司对经济类合同的管理，为公司管理带来方便性。

系统设置功能，可对操作人员进行权限设置，限制其访问的内容；合同管理功能，可方便公司各部门对合同的起草、审批、存档、查询；系统连接 Internet，方便用户在任何地方审批合同。

五、参考文献

[1].张岭,宋坤,梁冰.Delphi 程序开发典例宝典,2006.7.

[2].罗斌.Delphi for.NET 编程实例精粹,2006.5.

[3].明日科技,梁冰,宋坤.Delphi 范例完全自学手册.

六、进度计划

第 1 周——第 6 周：毕业实习，开题，查询相关资料。

第 7 周——第 8 周：应用分析、应用设计(开发环境、开发工具的确定熟悉开发环境及工具、

功能模块设计、代码设计)。 第 9 周——第 13 周：程序开发(编程及调试)。 第 14 周——第 16 周：论文书写及答辩。			
指导教师签字		时　间	年　月　日

摘　要

MIS管理信息系统，在强调管理，强调信息的现代社会中，MIS包含了众多学科，比如管理学、经济学、统计学以及计算机科学等。以这些学科为基础完成信息收集和加工，形成一个完善的系统。

而本文章讲的是合同管理系统，之所以做这样一个系统是因为以往合同都是通过手工管理，因为合同量大所以手工管理存在相当大的弊端，给企业带来诸多不便。本合同管理系统采用Delphi语言Delphi2009的编译环境下编写，满足各类经济类合同的管理界面友好操作方便。本系统以独特的内存流设计为亮点，流畅地完成了合同数据的起草、保存、调入、查看、修改等一系列功能；本系统有合同管理和系统设置两大模块，针对合同起草、提交、保存一整套管理流程以及操作人员和系统基本设置修改。系统经反复测试用户体验良好，系统稳固性强，设计合理满足用户需求。

关键词：合同管理　MIS　Delphi

Abstract

MIS is a information manage system, it is becoming more and more popular in the modern society of emphasize the management, emphasize the information. MIS contains many disciplines, such as management, economics, statistics and computer science, etc. On the basis of these subjects to complete the information collection and processing, and forming a perfect system.

And this article is about the contract management system, to do such a system because of previous contract is through the manual management, because of large amount of the contract so the disadvantages of manual management is quite big, bring so much inconvenience. This contract management system using Delphi language writing Delphi2009 compiler environment, meet all kinds of economic contract management friendly interface is easy to operate. With the unique memory flow design this system, smoothly completed the contract drafting, data storage, access, view, modify, and a series of features; This system has a contract management and system Settings module, against the contract draft, submit, save a set of management process and basic setup operators and system changes. System through repeated test of the user experience is good, strong stability, reasonable design to meet user needs.

Keywords: Contract Management MIS Delphi

目　录

毕业设计成绩单　I
毕业设计任务书　II
毕业设计开题报告　III
第 1 章　绪 论 1
1.1　课题背景及研究意义　1
1.2　国内外研究现状　2
1.3　课题研究内容 2
1.4　论文组织结构 3
第 2 章　系统开发工具 4
2.1　Delphi 语言简介　4
2.2　EjunGrid 简介 4
第 3 章　系统需求分析及概要设计　6
3.1　需求分析　6
3.2　可行性分析　7
3.2.1　技术可行性　7
3.2.2　经济可行性　7
3.2.3　社会因素可行性 7
3.3　系统的概要设计　8
3.3.1　系统结构设计　8
3.3.2　系统功能模块划分　8
3.3.3 工作流程 9
第 4 章　系统详细设计与实现　11

4.1 系统登录功能界面模块 11

4.2 系统主界面模块 14

4.3 合同管理功能模块 14

4.4 自定义内存流 25

4.4.1 内存流设计概要 26

4.4.2 内存流设计算法 27

4.5 系统分析结果测试 28

第 5 章 结论及展望 30

5.1 本文总结 30

5.2 展望 31

参考文献 32
致 谢 33
附录 A 34

英文原文 34

中文译文 39

附录 B 43

第 1 章　绪 论

1.1　课题背景及研究意义

随着时代的发展计算机已经应用到生活的各个方面。然而现在许多企业对于合同的管理还停留在手工操作，这大大地降低了企业的工作效率。近年来，随着公司交易的日益增多，合同信息量也在不断地增大[5]。随之而来的是管理市场工作日趋复杂繁重，要耗费大量人力、物力，而现有信息的管理水平不高，一直以来人们使用传统人工的方式管理信息。

采用手工管理合同，由于涉及的部门众多，需要管理的合同要素也各不相同，因此造成信息不集中，实时性不强，导致各部门协作，业务流程组建，监控制度执行方面效率不高，费时费力等问题，具体表现在如下方面：

文档管理困难：传统纸质合同与电子版合同共存，但对于不同的人员想阅读参考合同时，存在查找不方便的问题。尤其是领导需要了解合同文本时需要耗费很多时间。

进度控制困难：由于合同数目多，参与人员多，合同进度的控制基本靠手工和普通 word、excel 管理已很难满足公司发展需要，并且当领导想全局或全程了解合同情况时存在很大障碍。财务人员的付款依据也与进度密切相关，但同样存在障碍。

信息汇总困难：采用手工或 EXCEL 管理时，由于不同部门的数据格式不统一，采集也不能够及时继续，汇总工作需要耗费大量时间还不一定准确。对于领导的决策时间有一定的影响。

缺少预警机制：缺少对合同进度、结款等关键节点的预警，不能准确地预测近期可能的收支项目，不能帮助公司进行财务规划，掌控现金流，更好地发挥资金运作。

作为计算机应用的一部分，使用计算机对公司企业合同进行管理，具有着手工管理所无法比拟的优点. 例如:检索迅速、查找方便、可靠性高、存储量大、保密性好、寿命长、成本低等。这些优点能够极大地提高信息管理的效率，也是企业的科学化、正规化管理，与世界接轨的重要条件。因此，开发这样一套管理软件成为很有必要的事情。

1.2　国内外研究现状

规范基础数据管理，合同管理系统采用集中的数据管理，可以有效地改变企业信息分布比较散乱的现状。同时，系统提供自定义字段功能，可以为企业实现完整、规范的数据管理提供一个良好的平台支撑[12]。

提高管理效率，合同管理系统避免了手工管理合同出现，利用审批流管理，可以方便、快捷地处理企业管理事务；预警平台可以协助合同管理人员更轻松地

应对日常管理事务；报表管理的灵活定义，为管理部门的统计分析提供强大的技术支持。

实现标准化管理和个性化管理的有机结合，合同管理系统通过全局设置和权限分配，可以由公司制订统一采用的管理标准，比如，由公司设置业务单据必填字段，信息录入必须填写这些信息；数据字典可以为某些信息项提供标准的选择项，防止录入人员的录入随意性；设置统一的报表格式，信息输出时可以保证一致性。同时，系统也允许用户根据自身的实际情况采取个性化的措施，比如网格的列显示、自定义报表等。

为领导决策提供准确及时广泛的信息，利用互联网，领导可以在任何时间、任何地方进入系统，随时查阅与合同管理相关基本信息，准确定位下属人员的工作情况，掌握合同执行现状，并方便地生成各种数据报表或图表。同时，系统还提供强大的管理工具模块供客户选择，对这些信息进行深入分析，为公司决策人员提供更多的智力支持[10]。

为用户提供全面合同管理解决方案，合同管理系统与企业管理咨询相结合，可以为客户提供全面的合同管理解决方案。比如，通过系统的开放性和灵活性，可以有效地固化专业咨询的工作成果，这就为客户合同管理能力及水平的提升提供足够的平台支持。

总之，通过合同管理系统的应用和实施，可以为合同管理信息化和合同管理能力提升提供一个最佳的结合点，切实打造企业核心竞争力。

1.3 课题研究内容

本合同管理系统按照软件工程的方法进行需求分析与设计，根据科学化、系统化、信息化的合同管理原则进行设计[9]，系统分为九个主要功能模块：

单位设置：对签约单位基本属性的设置，其中包含单位编码、单位名称、单位机构代码、单位账号等等。

增加合同：增加合同的基本属性，资金计划，合同执行情况，合同起草及备注。

修改合同：对已经起草的合同进行修改，更新。

合同类别设置：增加删除合同类别，可导入导出 EXCEL 文件

权限设置：增加删除操作人员并设置操作人员的操作权限；

修改密码：修改当前操作员登录密码及基本信息；

端口设置：网络连接设置；

设置时间：系统时间设置；

升级：导入系统升级文件。

这些模块基本上满足了用户在合同管理方面的需求。实现了对合同的起草、

签约、修改和保存，以及合同从签约到执行再到最后完成等跟踪操作，操作人员信息的注册密码修改等功能。本管理系统的开发基于 Delphi2009 的开发环境，本着科学化、规范化、系统化的原则，并考虑到合同的实际情况，具有查询方便、安全保密性好、用户界面友好、容易操作等优点。

1.4 论文组织结构

本课题主要通过信息系统自动化技术对企业合同进行统筹管理，避免了以往手工管理合同带来的不便，降低管理成本，提高企业工作效率。本文的体系结构设计如下。

第 1 章：绪论。主要介绍合同管理的课题背景及研究意义、国内外研究现状、本课题的组织结构，使读者了解国内外合同管理的发展现状、优缺点、特色以及合同管理系统的信息化、自动化势在必行。

第 2 章：将简单介绍系统设计和开发过程中用到的主要工具和技术系统开发工具,包括 Delphi2009 以及相关插件。其中，Delphi 是进行系统开发的主要语言，Delphi2009 为系统开发的主要软件。

第 3 章：会介绍合同管理系统的需求分析、系统概要设计。需求分析部分介绍了系统的功能需求和性能需求详细。系统概要设计部分介绍了系统各模块的功能设计。

第 4 章：详细设计介绍了系统的登录主界面、各个功能管理模块。各个模块分别介绍了功能的设计与实现、核心代码的展示以及功能实现相应的图片示例。

第 5 章：结论及展望。对本文的结论进行总结，总结合同管理系统的各功能完成情况以及展望进一步的研究方向。

第 2 章　系统开发工具

2.1　Delphi 语言简介

Delphi 是 Borland 公司研制的新一代可视化开发工具，可在 Windows3.x、Windows95、WindowsNT、WindowsXP、Windows Vista、Windows7 等环境下使用。当前,Delphi 也可以在 LINUX 平台上开发应用,其在 LINUX 上的对应产品 Kylix[2]。Delphi 拥有一个可视化的集成开发环境(IDE)，采用面向对象的编程语言 ObjectPascal 和基于部件的开发结构框架。Delphi 它提供了 500 多个可供使用的构件，利用这些部件，开发人员可以快速地构造出应用系统。开发人员也可以根据自己的需要修改部件或用 Delphi 本身编写自己的部件[9]。

Delphi 具有简单、高效、功能强大的特点,被称为第四代编程语言。和 VC 相比,Delphi 更简单、更易于掌握,而在功能上却丝毫不逊色;和 VB 相比,Delphi 则功能更强大、更实用。可以说 Delphi 同时兼备了 VC 功能强大和 VB 简单易学的特点。它一直是程序员至爱的编程工具。Delphi 具有以下的特性：基于窗体和面向对象的方法，高速的编译器，强大的数据库支持，与 Windows 编程紧密结合，强大而成熟的组件技术。但最重要的还是 Object Pascal 语言，它才是一切的根本。Object Pascal 语言是在 Pascal 语言的基础上发展起来的，简单易学[6]。

2.2　EjunGrid 简介

EjunGrid 是一款类似 Excel 风格的高品质表格控件,我们设计的目标是让广大软件开发者能够轻松快速开发出专业、高水准的软件产品，使 您的软件具备方便快捷的录入界面、清晰漂亮的数据显示界面、完美强大的打印预览功能、可以让您的用户在打印预览时实时方便的调整页面布局，所见即所得，操 作方式与 Excel 完全兼容，输出的报表精美典雅[5]。

众多优质的功能，让 EjunGrid 跻身于高端表格控件之列，EjunGrid 是纯 Delphi 表格控件，同时提供 Web 插件版，用于开发 Web 报表，ActiveX 插件版用于 VB，VC 等工具开发。我们设计的目标是让广大软件开发者能够轻松快速开发出专业、高水准的软件产品,，使您 的软件具备方便快捷的录入界面、清晰漂亮的数据显示界面、完美强大的打印预览功能、可以让您的用户在打印预览时实时方便的调整页面布局，所见即所得，操作 方式与 Excel 完全兼容，输出的报表精美典雅。

主要特点

强大的单元格合并功能，客户区、 表头、列头，都可以随意合并单元格，能够制作出任意复杂的表格 ；

支持行锁定和列锁定,拖动滚动条时固定行和固定列不随滚动条滚动而改变位置，适合显示商品名称、编号等固定信息;

单元格可以插入任何类型的对象 ；

兼容 Excel 操作方式，使您的软件用户能够轻易上手，减少培训费用 ；

支持 Excel 方式的拖动选择， 拖动复制，行选， 列选 ；

能够和 Excel 一样，拖动选择框右下角的小方框进行行填充和列填充；

可以和 Excel 之间相互复制粘贴内容 ；

可以灵活地控制选择框的运行轨迹，例如用户在第一列输入完数据后按回车键，您可以根据需要让选择框掠过第二列直接跳转到第三列，或者您需要的任何地方 ；

丰富鼠标事件和键盘事件，完善的开发接口 ；

可根据页面宽度按比例自动拉伸列宽；

可根据页面高度自动插入空白行充满整个页面高度；

可随意选择打印范围，打印表格中指定的区域；

更强大的是：可以指定表格中的某些行和列为标题行和标题列，打印时每页都出现。这样可以轻松打印出每页都需要的表头或列头；

可以在打印预览时拖动鼠标调整页边距、行高、列宽，调整时以虚线提示调整的位置，所有操作完全适应 Excel ；

可以选择预览调整的结果是否实时同步到表格中；

可以设置多行页眉页脚，自动选择打印页码、总页数、日期、事件等等，可以设定字体颜色 ；

可以设置多行标题，实现主大标题、副标题等效果[8]。

第 3 章　系统需求分析及概要设计

3.1　需求分析

采用手工管理合同，由于涉及的部门众多，需要管理的合同要素也各不相同，因此造成信息不集中，实时性不强，导致各部门协作，业务流程组建，监控制度执行方面效率不高，费时费力等问题，具体表现在如下方面：

文档管理困难：传统纸质合同与电子版合同共存，但对于不同的人员想阅读参考合同时，存在查找不方便的问题。尤其是领导需要了解合同文本时需要耗费很多时间。

进度控制困难：由于合同数目多，参与人员多，合同进度的控制基本靠手工和普通 word、excel 管理已很难满足公司发展需要，并且当领导想全局或全程了解合同情况时存在很大障碍。财务人员的付款依据也与进度密切相关，但同样存在障碍。

信息汇总困难：采用手工或 EXCEL 管理时，由于不同部门的数据格式不统一，采集也不能够及时继续，汇总工作需要耗费大量时间还不一定准确。对于领导的决策时间有一定的影响。

缺少预警机制：缺少对合同进度、结款等关键节点的预警，不能准确地预测近期可能的收支项目，不能帮助公司进行财务规划，掌控现金流，更好地发挥资金运作。

基于手工管理的种种缺点，为了更好地管理合同，实现管理自动化，我们有必要将合同管理的流程与现代的计算机技术相结合，简历合同信息管理系统，以便于合同管理的顺畅进行，实现合同管理流程全过程的电子化操作。通过与企业的管理人员与操作人员进行细致交流，最终确定本系统要具备以下功能：

合同管理模块

单位设置：对签约单位基本属性的设置，其中包含单位编码、单位名称、单位机构代码、单位账号等，遍历起草合同时增加的来往单位基本信息，并对其进行修改保存。

增加合同：起草合同的首页基本属性，资金计划，合同执行情况，合同文本，合同执行来往单位及备注。

修改合同：对已经起草的合同进行修改，更新，从目录遍历已经起草完成的合同对合同状态进行逐级提交，签约-执行-完成。

合同类别设置：增删改查合同类别，方便合同起草修改时对类别的调入，可导入导出 EXCEL 文件

系统设置模块

权限设置：增加删除操作人员并设置操作人员的操作权限；

修改密码：修改当前操作员登录密码及基本信息；

端口设置：网络连接设置，端口、ip 地址的修改；

设置时间：系统当前时间设置；

升级：导入系统升级文件。

3.2 可行性分析

目前及至将来，企业竞争将主要在智能化，信息化方面展开，企业管理信息化的发展势在必行。合同管理系统可以为企业管理者提供合同管理方面及时而准确的信息，并且可以对合同的起草、执行、和完成方面提供控制。

3.2.1 技术可行性

在Windows环境下搭建Delphi开发平台，整个系统用Delphi语言设计开发；数据保存采用数据-结构体-内存流-本地文件的方式保存；数据调用采用本地文件-内存流-调入记录-表格的方式调用已经保存的数据。网络连接采用

3.2.2 经济可行性

从项目开发的投入金额、收益、收益投资比、投资回收周期和敏感性方面来考虑。在项目开发的投入上，开发和使用合同信息管理系统需要一台PC机，在软件上，需要Delphi09。

当前合同的管理比较散乱，信息不及时，大部分工作需要手工操作，对信息的维护不仅费用高、工作量大，而且对信息的安全性很难得到保障。而该合同管理系统只需支出适量的资金进行本系统的开发。使用本系统后大部分工作实现自动化，企业只需花费很少的人力和物理进行系统的维护即可，并且降低了数据被无意破坏的风险。

3.2.3 社会因素可行性

从法律因素看，本系统开发所使用的软件都是正版的，所有的技术资料都是由单位保管的，另外将通过签订合同来确定开发单位和使用单位的职责和违约责任，因此，开发合同管理系统是可行的。

从用户使用方面看，本系统的一般操作人员和系统管理员要求有一定的计算机基础和一定的计算机专业知识。系统的操作人员经过简单培训将会熟练地使用本系统。因此在用户使用方面，开发合同管理系统是完全可行的。

3.3 系统的概要设计

3.3.1 系统结构设计

本系统主要分为以下几个子模块：

单位设置：对签约单位基本属性的设置，其中包含单位编码、单位名称、单位机构代码、单位账号等等。

增加合同：增加合同的基本属性，资金计划，合同执行情况，合同起草及备

注。

修改合同：对已经起草的合同进行修改，更新。

合同类别设置：增加删除合同类别，可导入导出 EXCEL 文件

权限设置：增加删除操作人员并设置操作人员的操作权限；

修改密码：修改当前操作员登录密码及基本信息；

端口设置：网络连接设置；

设置时间：系统时间设置；

升级：导入系统升级文件。

系统结构设计图如图3-1所示。

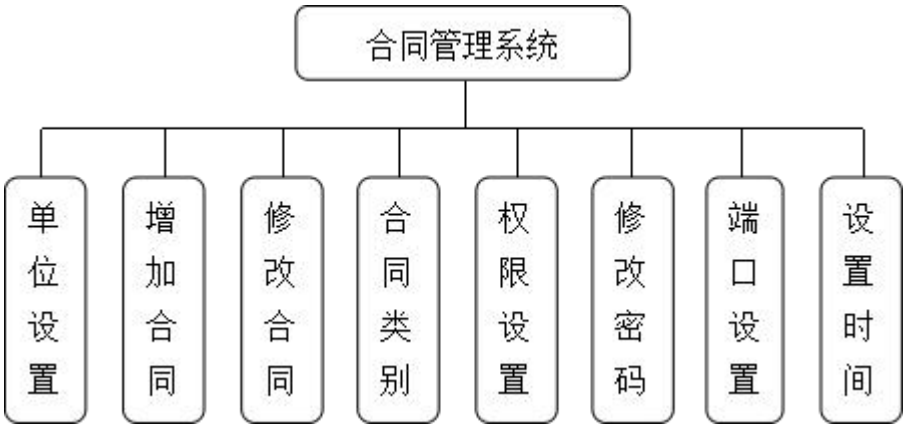

图 3-1 系统结构设计图

3.3.2 系统功能模块划分

本系统的主要功能模块划分如图3-2所示。

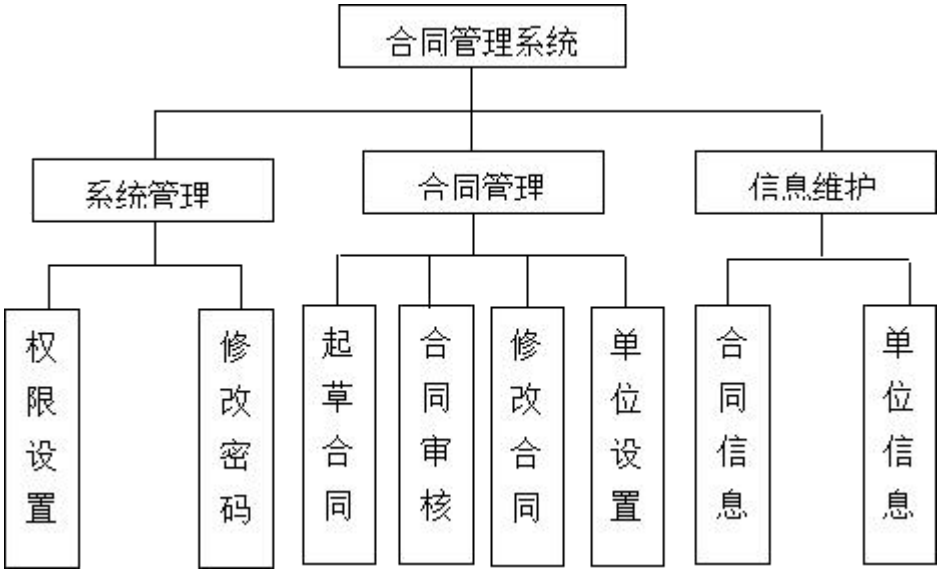

图3-2 系统模块图

(1) 权限设置

增加删除操作人员并设置操作人员的操作权限；

修改密码

修改当前操作员登录密码及基本信息；

起草合同

增加合同的基本属性，资金计划，合同执行情况，合同起草及备注；

合同审核

相关部门对合同进行审核；

修改合同

对已经起草和签约的合同进行修改，更新；

单位设置

对签约单位基本属性的设置，其中包含单位编码、单位名称、单位机构代码、单位账号等等；

合同信息

对已经保存的合同信息进行管理维护更新；

单位信息

对已经保存的单位信息进行管理维护更新。

3.3.3 工作流程

本系统由起草合同开始，各个部门逐级提交，通过每个部门的审核、审批方可执行；如果合同不符合某个部门的要求，该部门可以执行回退，合同退到上一个部门进行修正，直到达标后方可再向上级提交。

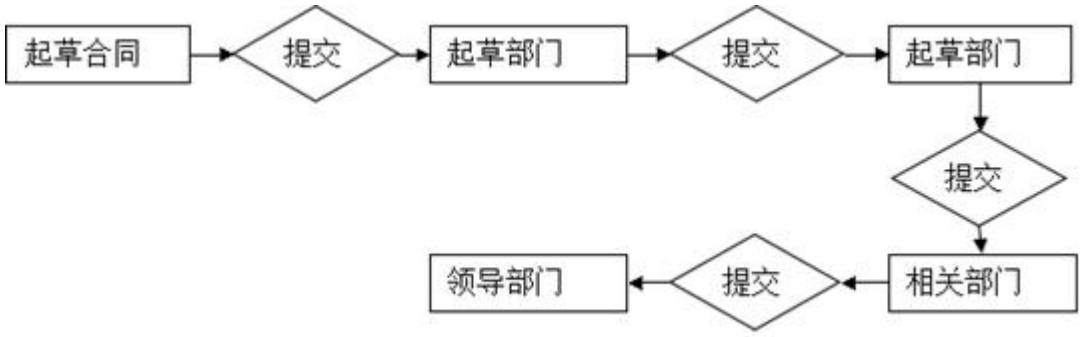

图3-3 系统提交工作流程图

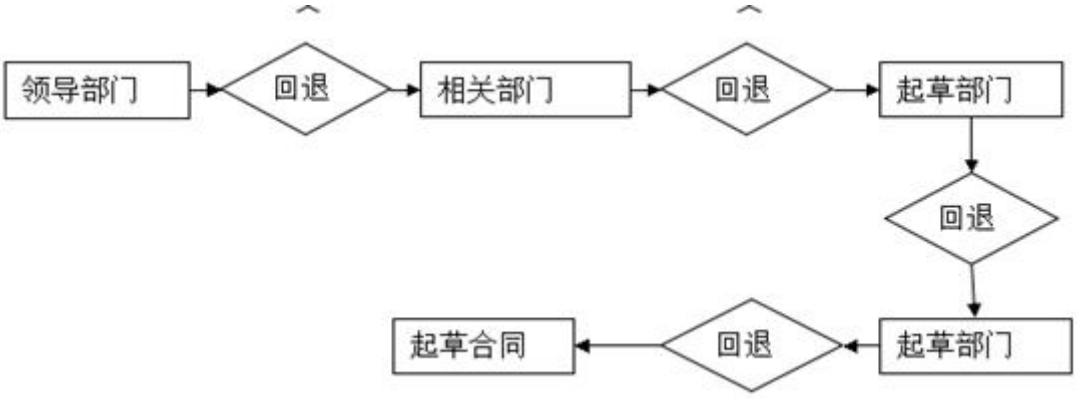

图3-4 系统回退工作流程图

第 4 章　系统详细设计与实现

4.1　系统登录功能界面模块

当启动本系统时，首先要求用户进行登录，用户登录模块实现了用户名和密码与数据中User表中的登录名和密码的验证工作，并且将每次用户登录的情况记录到事件日志中，登录窗体设计界面如图4-1所示。

图4-1 登录界面

当用户输入用户名和密码时，单击“登录”按钮，将开始验证用户，并根据验证的结果决定是否进入合同管理系统，同时根据用户的权限设置不同，主窗口上的菜单栏也会不一样。用户登录流程如图4-2所示。

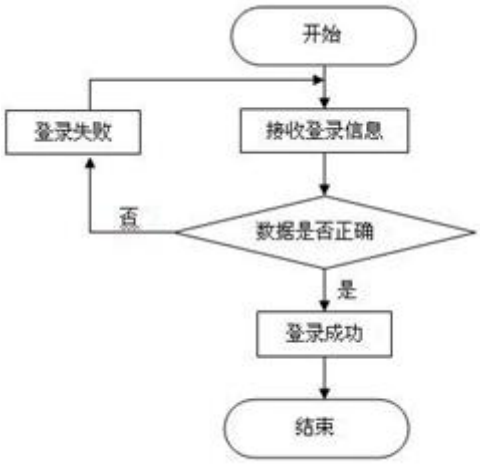

图4-2 登录流程

登陆操作实现了密码和用户名以及用户所属单位的匹配，密码、用户名及单位信息通过内存流存交由一个指定的文件保存，匹配时锁定到此文件对文件内的存储信息进行遍历匹配，匹配成功则登陆系统主界面并且记录用户名和对应单位，下次登陆时直接调用方便二次登陆，登录失败显示错误信息，提示重新登录；系统刚登陆时通过调用系统时间可以实现以不同季度的时间登陆，录入此季度的数

据。其主要代码如下：

```
procedure TDebarkationForm.BitBtn1Click(Sender: TObject);
begin
    My_Bz := -1 ;
    aa :=   RichEdit1.ItemIndex ;
    My_Memory := TmemoryStream.Create ;
    if   aa >= 0 then
    begin
    //单位记录结构用来记录匹配成功的单位信息，方便下次登陆直接调入
    My_Data_Module.My_curr_Dw_JiGou:=
My_Data_Module.Debarkation[aa ];

My_Data_Module.My_SS_Dw_JiGou:=My_Data_Module.My_curr_Dw_JiGou;
            if   My_Data_Module.My_SS_Dw_JiGou.Sx > 103   then
            begin
                k1 := My_Data_Module.My_SS_Dw_JiGou.Level -1 ;
                for   level := 0 to   k1   do
                begin
                   k := My_Data_Module.My_SS_Dw_JiGou.Parent;

My_Data_Module.My_SS_Dw_JiGou :=My_Data_Module.Debarkation[ k ];
                    if   My_Data_Module.My_SS_Dw_JiGou.Sx <= 103      then
                    begin
                       Break ;
                    end;
                end;
             end;
            / /调入系统时间
            MiMaQR.ND :=   My_Data_Module.My_Curr_SystemTime.Year;
            //路径
            StrCopy( MiMaQR.My_Path,Pchar(Copy(
                            My_Data_Module.My_Curr_Path.Curr_Nd_Path,
                            1,
                            High(MiMaQR.My_Path) )));
```

```
            //单位编码
            StrCopy( MiMaQR.DWQMBM,Pchar(Copy(Str, 1,
                                    High( MiMaQR.DWQMBM ) )));
            //用户名
            StrCopy( MiMaQR.YHM,Pchar(Copy(ComboBox1.Text,1,
                                      High( MiMaQR.DWQMBM ) )));
```

//匹配用户名和所属单位，匹配成功登陆系统并且记录用户名和单位信息,匹配失

败显示错误信息，提示重新登陆。

```
                if ( MyDebarkation.Sx = 106 )   and
                          (MyDebarkation.Mc = ComboBox1.Text)   and
                                    (MyDebarkation.MiMa = RichEdit2.Text)
                then
                begin
                  //操作人员记录结构
                  My_Data_Module.My_curr_Ry_JiGou                    :=
MyDebarkation ;
                  My_Data_Module.My_curr_Ry_JiGou_Zz   := k;
                  My_Dw_QBM := GetFullBm(aa);
                  //生成选择树
                  My_Make_Tree(   aa    ) ;
                  My_Bz:=1;
                  //存入当前登陆用户名文件
                  SaveDebarkaUserName( );
                  Close;  exit;  end;
              end;
            end;
        end;
        ShowMessage('用户名或操作员或密码错误，请检查，谢谢！');
        RichEdit2.SetFocus();
        My_Memory.free;
    end;
```

4.2　系统主界面模块

主窗口的用户界面由Panel，TFLButton，TTimer，TEjunLicens，TEdit，TBitBtn

控件组成，为了控制不同权限的用户所能操作的菜单项和按钮，在窗体加载时，将根据不同的权限来显示和隐藏菜单项和按钮，存在User表中的权限字段中，例如Admin的权限字段保存了权限设置，导入升级文件，单位设置等值，它们以冒号分隔的形式保存在字段中，当用户通过验证时，分解权限列表，遍历菜单栏，将权限字段中所对应的菜单选项的Visible设为True,否则将Visible选项设置为False,从而使得不同用户登录界面后具有不同的操作权限。主界面如图4-3所示。

图4-3 主界面设计

4.3 合同管理功能模块

合同管理主要功能分为6个子模块：起草合同、修改合同、部门审核、部门审批、单位设置和合同类别设置。这些功能位于主窗口的“基本文件”菜单选项下，可通过系统设置中的权限设置，将每个模块分别成可操作和拒绝操作的状态，其各个模块功能如下所示。

1. 起草合同：起草合同的基本属性，资金计划，合同执行情况，合同文本，合同来往单位设置及备注。

（1）合同首页：记录合同的基础属性。起草部门、合同编号、合同名称、合同属性、合同类别、合同有效期、起草部门意见、财务部门意见、财务部门意见、律师部门意见、领导签字、备注。每个部门中都会有相应的复核人、负责人和经办人的基本信息，签字意见等，且由系统自动调入人员信息，不可手动输入。例如起草部门就是当前登录系统的部门，起草部门的经办人就是当前的起草人员自动调入，负责人在部门审批通过时调入，复核人在部门审核通过时调入，同时经办人、负责人、复核人的下方还会调入当前系统时间，用来记录起草的准确时间；合同属性采用下拉框样式，分为收款合同和付款合同；合同类型同样采用下拉框样式，类型在合同类别模块中设置；此界面包含打印、打印预览、提交、确

定、退出功能，采用TButton按钮控件进行设计。合同首页界面如图4-4所示。

图4-4 合同首页界面

核心代码如下：

/*起草部门,用My_Data_Module.My_JiGouQBM_Get函数得到起草部门的部门编
码和部门名称*/

```
My_Temp_Grid.Cells[Ej_HTSY_QCBM_NR_COL,Ej_HTSY_QCBM_NR_ROW].
Text :=

My_Data_Module.My_JiGouQBM_Get(My_Data_Module.My_XzDanWei_Zz)
                                +':'
+Trim(My_Data_Module.My_curr_Dw_JiGou.Mc);
```

/*经办人，调用My_Data_Module.My_curr_Ry_JiGou.Mc储存的名称属性，复制到FL_HTSY_JG.QCBMYJ_JBR，其中FL_HTSY_JG是合同首页表格建立时共同创建的合同首页的结构体，记录了合同首页表格中所有属性*/

```
        StrCopy( FL_HTSY_JG.QCBMYJ_JBR    ,
                PChar(Copy(
                    My_Data_Module.My_curr_Ry_JiGou.Mc,
                    1 ,
                    High(FL_HTSY_JG.QCBMYJ_JBR))) ) ;
    //联系电话
        StrCopy( FL_HTSY_JG.QCBMYJ_LXDH    ,
                PChar(Copy(
                    My_Data_Module.My_curr_Ry_JiGou.SJH,
```

```
                    1 ,
                    High(FL_HTSY_JG.QCBMYJ_LXDH))) ) ;
    //经办人,将合同首页结构体中的属性调入到表格

Ej_HTSY.Cells[Ej_HTSY_QCBMYJ_JBR_NR_COL,Ej_HTSY_QCBMYJ_JBR_NR_ROW].
                    Text := Trim(FL_HTSY_JG.QCBMYJ_JBR);
```

（2）合同资金计划：记录合同计划资金和已完成资金项目。本模块包含以下内容：合同编号、合同名称、合同属性、合同总额、合同工期、收还款日期、收还款计划、归还额度、归还日期、记账凭证号、财务确认、未归还额度、违约金、经办人、审核人、领导签字、备注。其中合同编号、合同名称、合同属性调用合同首页的对应属性，自动生成，采用TEdit文本框控件进行设计；当合同属性为收款合同时，对应合同资金计划模块中的收款日期和收款计划，同理当合同属性为还款合同时,对应合同资金计划模块中的还款日期和还款计划；第一行为合计通过计算得出不能手写，其计算公式为：合计=收款额度+调整=归还额度+未归还额度，每一个属性发生变化合计单元中的数值都会进行相应的调整，最终满足上述公式的计算。此界面包含增加行、删除行、打印、打印预览、提交、保存功能，采用TButton按钮控件进行设计。合同资金计划界面如图4-5所示。

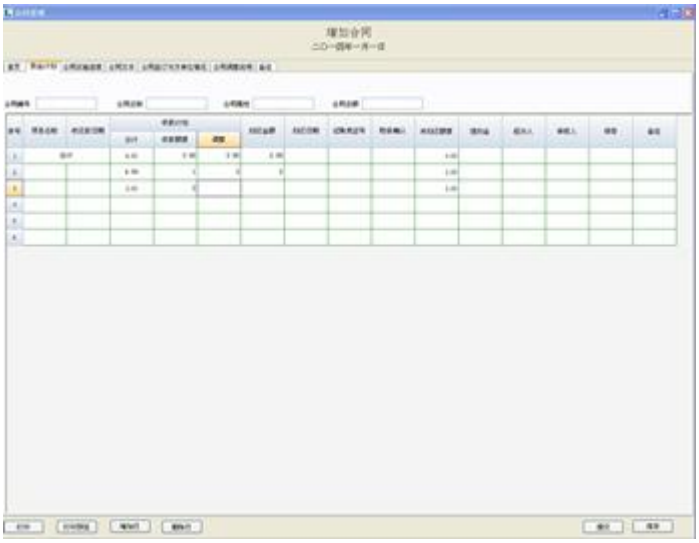

图4-5 合同资金计划界面

下面介绍模块中用到的核心代码，合计=收款额度+调整=归还额度+未归还额度，每一列的合计统计到第一行，每一行的合计统计到第一列，每次计算的得到的数据通过实时赋值函数马上显示到表格，方便用户实时比对实时查看，减少出错率。其代码具体实现如下：

```
  //当前单元实时赋值
 Ej_ZJJH.Cells[Ej_ZJJH.CurCol,Ej_ZJJH.CurRow].Text:=
       Ej_ZJJH.EditorText;
  //合计=收款额度+调整=归还额度+未归还额度
  begin
    //收款额度
    sum    := MyStrToCurr(Ej_ZJJH.Cells[Ej_ZJJH_SHKED_COL,i].Text);
    shoukuanedu:= shoukuanedu + sum;
    //调整
    sum1 :=     MyStrToCurr(Ej_ZJJH.Cells[Ej_ZJJH_TZ_COL,i].Text);
    Tiaozheng:= Tiaozheng + sum1;
    //合计=计划收款额度+调整
    sum :=    sum1 + sum;
    Ej_ZJJH.Cells[Ej_ZJJH_HJ_COL,i].Text :=    MyCurrToStr(sum);
    Zongji:= Zongji + sum;
    //归还额度
    sum1 := MyStrToCurr(Ej_ZJJH.Cells[Ej_ZJJH_GHED_COL,i].Text);
    huankuan :=     huankuan +    sum1;
    //未归还额度=合计-归还额度
    sum :=     sum - sum1;
    Ej_ZJJH.Cells[Ej_ZJJH_WGHED_COL,i].Text :=    MyCurrToStr(sum);
    Weiguihuan :=    Weiguihuan + sum;
end;
```

（3）合同实施进度：记录合同目前的进度。合同编号、合同名称、合同属性、合同总额、项目名称、起始日期、结束日期、预计完成情况、到期完成情况、验收部门、验收人、负责人、验收日期、备注。其中合同编号、合同名称、合同属性调用合同首页的对应属性，自动生成，采用TEdit文本框控件进行设计；起始日期、结束日期、审批日期的选框通过点击弹出日历框选择日期。此界面包含增加行、删除行、打印、打印预览、提交、保存功能，采用TButton按钮控件进行设计。合同实施进度界面如图4-6所示。

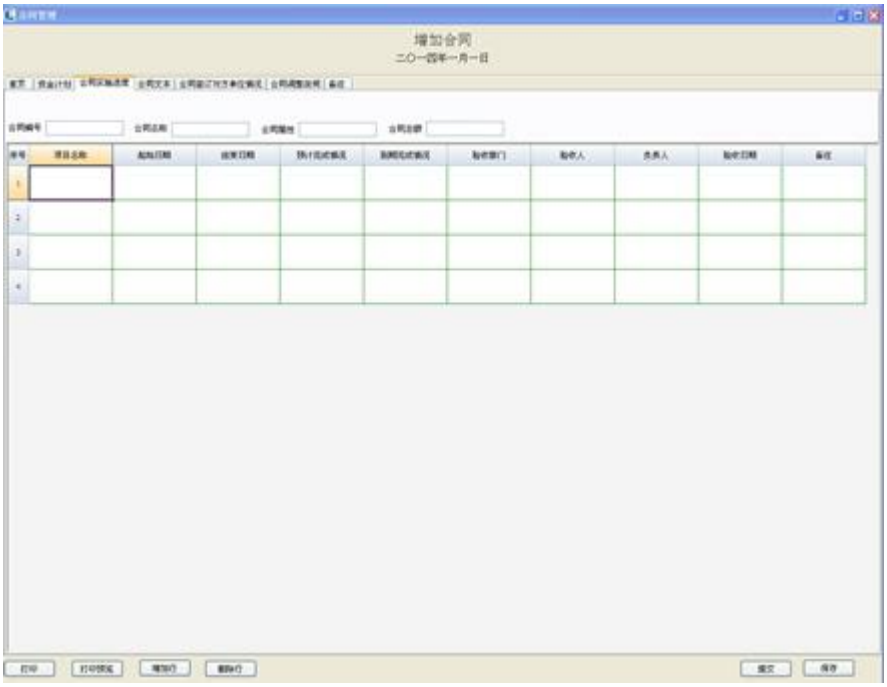

图4-6 合同实施进度界面

日历调用的核心代码设计如下：通过一个My_SJ参数的传递，将当前选中单元的日期temp_Grid.CurCell.Text传递给My_Calender_Form。其中My_Calender_Form是一个计算日期的控件，可以显示出2000年到2149年度的所有日期。

```
My_Calender_Form.My_BZ := 3 ;
        My_Calender_Form.My_SJ:=
MyYMDToJYSj(temp_Grid.CurCell.Text);
        My_Calender_Form.ShowModal();
        if   My_Calender_Form.My_BZ = 1 then
        begin
            temp_Grid.CurCell.Text :=
                        MyYMDToDx(My_Calender_Form.My_SJ,4 ) ;
```

（4）合同文本：记录合同的文本文件。此界面可以编辑文本文档，同时包含打印、打印预览、提交、确定、退出功能。合同文本和备注试用Delphi提供的文本编辑插件，其操作类似于office word的文本编辑。此界面包含打印、打印预览、提交、确定、退出功能，采用TButton按钮控件进行设计。合同文本界面如图4-7所示。

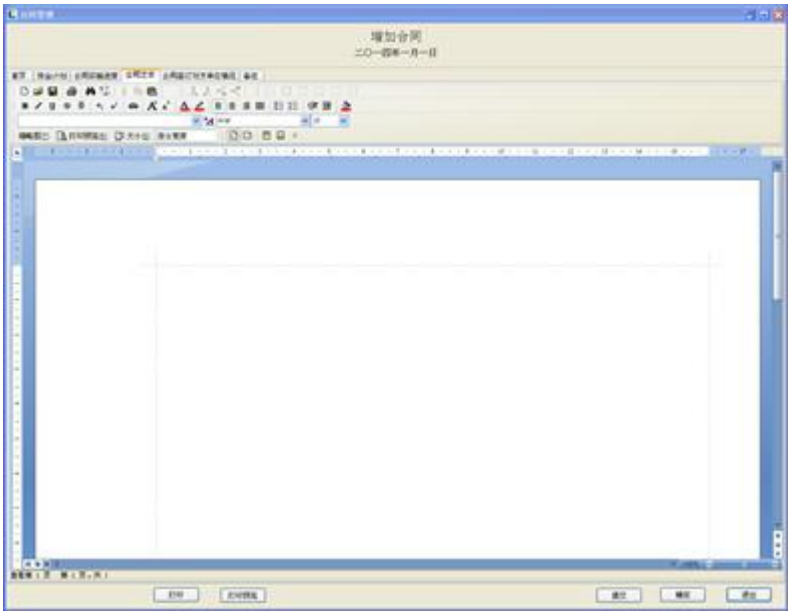

图4-7 合同文本界面

（5）合同签订单位情况：记录签订合同对方单位的基本信息。包括单位编码、机构代码、单位名称、地址、单位账号、账号开户行、法人、法人电话、联系人、联系人电话、传真、电子邮箱、注册资金、信誉度、备注。此界面包含打印、打印预览、提交、确定、退出功能，采用TButton按钮控件进行设计。合同来往单位界面如图4-8所示。

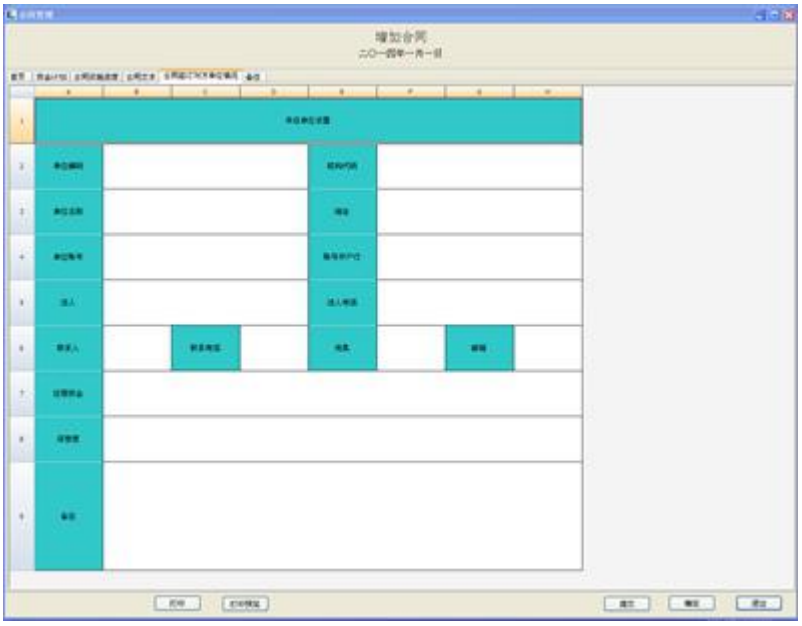

图4-8 合同单位界面

2. 修改合同：用于遍历和修改起草完成的合同。通过次界面的目录遍历已经保存成功的合同，可对执行、签约和完成状态的合同进行修改, 通过目录可以

看到所有已经保存的合同编号和合同名称，在目录中用鼠标左击要查看的合同，在右侧合同浏览的界面上会调出对应的合同信息，如果有错误可以在此界面修改，核对无误后可以选择保存、提交、打印等功能对合同进行操作，当合同内容发生变动后，点击左侧目录其他合同时系统会提示保存。此界面包含打印、打印预览、提交、确定、退出功能，采用TButton按钮控件进行设计。修改合同界面如图4-9所示。

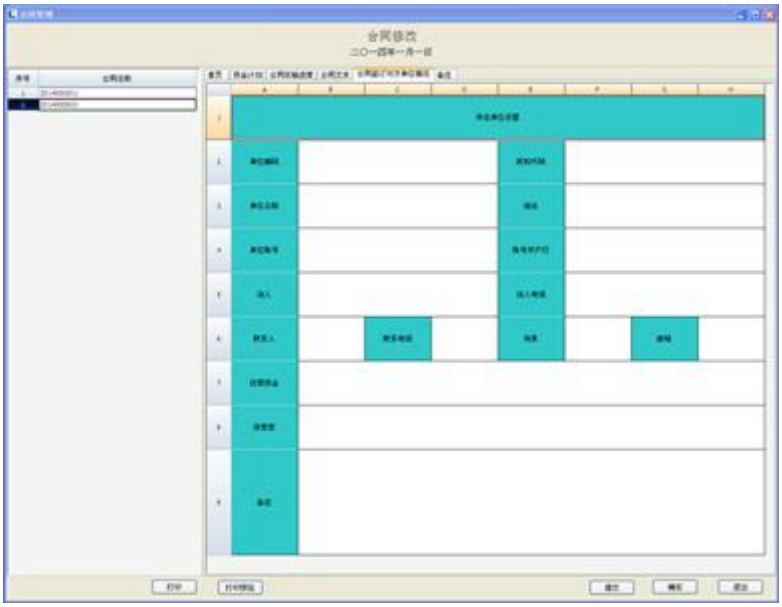

图4-9 合同修改界面

目录操作及合同编号生成的核心代码如下：合同编号由当前年度、My_Ws以及文件头长度组成，其中年度调用系统时间的年份，My_Ws是一个系统常量用来定义编号的长度其数值等于1000000，文件头长度是每保存一个合同都会向上叠加，记录了合同编号的顺序。如果合同内容在修改界面发生了变化，用old_ML来记录当前变化合同的所属目录，当点击其他目录时会触发数据保存的函数，询问是否保存已修改的合同，其中old_ML是一个全局变量，保证修改的合同保存到正确的位置。

```
//合同起草编号
My_HTML.HTQCBH := GongNeng_ZY.Nd * My_Ws + Info.FileLengthHead;
//调入目录函数
procedure THTGL_Form.DRML();
var
    i : Integer;
    hang : Integer;
begin
    My_File_Name := My_Data_Module.My_Curr_Path.Curr_Nd_Path    + '\' +
```

```
                                    //合同管理文件夹
                                    HTGLWJJ   + '\' +
                                    //合同文件夹
                                    HTWJJ + '\' +
                                    //合同目录前缀文件名
                                    HTMLQZ_WJM+
                                    MyIntToStr( My_SystemTime.Year );
        My_memo_ML.Clear;
        if     My_Read_Sever_File(
                                My_File_Name,//文件名
                                My_memo_ML,//文件内存流
                                0,//文件从0偏移量 -1:不偏移
                                -1,//文件长度 -1:取全部文件
                                //File_Mode : Byte; //1:正常读文件并关闭,2：
读文件并锁定
                                @i //文件句柄
                                 )  > 0          then
        begin
            My_memo_ML.Position := 0;
            My_memo_ML.ReadBuffer(FileInfo,SizeOf(FileInfo));
            Ej_HTML.RowCount :=   FileInfo.RecordCount+1;
            hang := 1;
            for i := 1 to FileInfo.RecordCount   do
            begin
                My_memo_ML.ReadBuffer(HTML,SizeOf(HTML));
                            Ej_HTML.RowCount :=   hang;
            end;
        end;
    end;
```

3. 单位设置：遍历修改来往单位的基本信息。包括单位编码、机构代码、单位名称、地址、单位账号、账号开户行、法人、法人电话、联系人、联系人电话、传真、电子邮箱、注册资金、信誉度、备注. 此模块调入合同起草时录入的来往单位信息，并且可对单位信息进行修改保存。单位设置界面如图4-10所示。

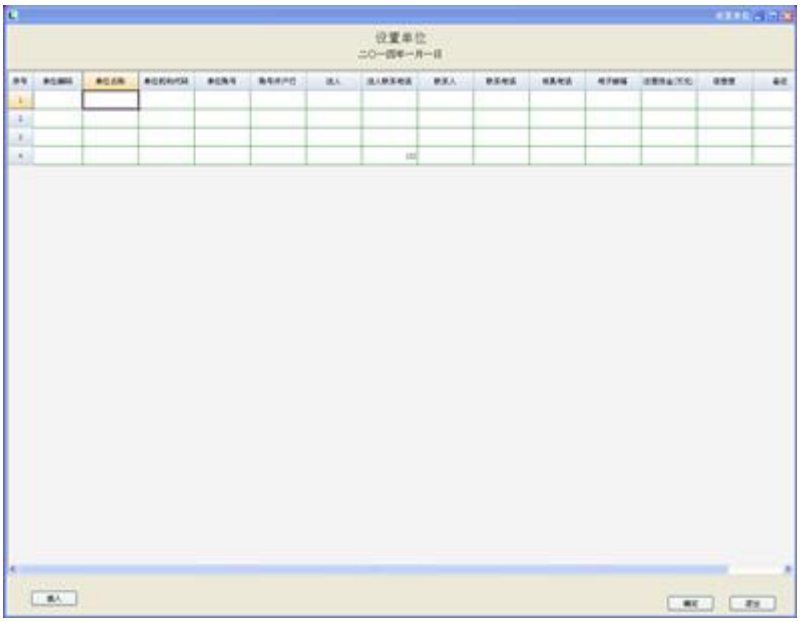

图4-10单位设置

4. 合同类别设置：对合同类别进行增删改查、排序、批量导入、存盘、打印。合同类别设置界面如图4-11所示。

图4-11合同类别设置界面

5. 部门审核：对起草完成的合同进行审核，只有审核部门有对其操作的额权限。增加了同意和拟同意两个按钮，如果起草合同完全没有问题，则由审核审核部门的操作人员点同意按钮，此时复核人的姓名则会被调入，复核人上方会显示同意字样；如果合同起草存在问题，则点击拟同意按钮，复核人姓名同样被调入，复核人上方显示拟同意字样，然后回退或者保存，等待后期的处理。审核完成后提交给部门审批，合同状态也有部门审核转变成合同审批，合同状态会在右上角显示。此模块主要包含打印、打印预览、同意、拟同意、提交、回退、保存功能，采用TButton按钮控件进行设计。部门审核界面如图4-12所示。

图4-12部门审核界面

6. 部门审批：对审核通过的合同进行审批批准通过后合同才可以执行，只有审批部门有对其操作的额权限。如果合同没有纰漏的话由审批人员点击同意按钮，负责人的姓名会被调入，负责人上显示同意字样，然后对合同进行提交操作可将合同提交给下一个部门进行审核、审批；如果合同存在问题则点击拟同意按钮，负责人姓名同样被调入，上方显示拟同意字样，然后回退或者保存，等待后期的处理。此模块主要包含打印、打印预览、同意、拟同意、提交、回退、保存功能，采用TButton按钮控件进行设计。部门审核界面如图4-13所示。

图4-13部门审批界面

合同提交流程的核心代码设计：采用一个全局变量My_BZ来标识合同的不同状态，My_BZ=1表示起草合同，My_BZ=2表示修改合同，My_BZ=3表示起草部门审核，My_BZ=4表示起草部门审批，以此类推到My_BZ=10表示领导部门审

批，这样设计方便对每个模块分开操作，每个模块看似不可分割但是对每一个的修改只需要用到一个case语句就可以实现互不影响的目的，而且也方便了日后程序的修改优化，添加删除一些模块或者功能的话只需要增加或者减少My_BZ。此部分设计主要代码如下：

```
        Case   My_BZ   of
         1: //   起草合同
         begin
           lblBT.Caption   := '增加合同' ;
           lblSJ.Caption:=
MyYMDToDx( My_Data_Module.My_Curr_SystemTime,1 );
             //经办人
             StrCopy( FL_HTSY_JG.QCBMYJ_JBR   ,
                    PChar(Copy(
                       My_Data_Module.My_curr_Ry_JiGou.Mc,
                       1 ,
                       High(FL_HTSY_JG.QCBMYJ_JBR))) ) ;
              //联系电话
             StrCopy( FL_HTSY_JG.QCBMYJ_LXDH   ,
                    PChar(Copy(
                       My_Data_Module.My_curr_Ry_JiGou.SJH,
                       1 ,
                       High(FL_HTSY_JG.QCBMYJ_LXDH))) ) ;
          end;
        2:   //  合同修改
        begin
           //调入函数
           DR();
           lblBT.Caption := '合同修改' ;
           lblSJ.Caption:=
MyYMDToDx( My_Data_Module.My_Curr_SystemTime,1 );
           btn_tijiao.Caption := '提交';
           BtnHT.Caption := '回退';
            //经办人
             StrCopy( FL_HTSY_JG.QCBMYJ_JBR   ,
```

```
                PChar(Copy(
                    My_Data_Module.My_curr_Ry_JiGou.Mc,
                    1 ,
                    High(FL_HTSY_JG.QCBMYJ_JBR))) ) ;
        end;
        3:      //部门审核
        begin
          //调入函数
          DR();
          lblBT.Caption   := '部门审核' ;
          lblSJ.Caption:=
MyYMDToDx( My_Data_Module.My_Curr_SystemTime,1 );
          btn_tijiao.Caption := '提交';
          BtnHT.Caption := '回退';
           //复核人
            StrCopy( FL_HTSY_JG.QCBMYJ_FHR   ,
                PChar(Copy(
                    My_Data_Module.My_curr_Ry_JiGou.Mc,
                    1 ,
                    High(FL_HTSY_JG.QCBMYJ_FHR))) ) ;
        end;
```

4.4 自定义内存流

内存流的设计是整个系统运行的支柱，内存流保证了系统中同一单元和不同单元之间数据的保存和访问，同一单元的数据通过内存流实现了数据的保存和调用，本单元的数据在其他单元可能会重复调用，不同单元的数据访问只需要通过内存流将路径定位到需要访问的文件，通过管道技术实现了数据的本地访问和网络位置访问，本系统采用文本到结构体到内存流，通过管道存入静态文件的方式，提高了系统的操作性，使系统方便移植。

4.4.1 内存流设计概要

1. 保存过程。因为合同的存储涉及到多张表格和文本的保存，所以本系统在初始化时创建两个个内存流，内存流甲和内存流乙，内存流甲用来搬运数据，内存流乙用来汇聚数据；每个表格和文本框创建的同时建立一个相应的结构体，每张表格和文本的存储过程都要经历录入数据—写入结构体—导入内存流—汇聚内存流，当内存流甲将数据成功汇聚到内存流乙后释放内存流甲，再创建新的内

存流甲进行数据的搬运；当数据全部都汇聚到内存流乙后，内存流乙会将数据送给管道函数然后存入目标文件，释放内存流乙。每次的数据保存不管是成功还是失败都会执行Clear()函数，对录入的数据进行清空工作，方便下次数据的录入。本系统内存流工作流程图如图4-14。

图4-14内存流设计流程图

2. 调入过程。数据的调入为保存的逆过程：文件—内存流—结构体—读出；调入数据之前需要进行文件查找，用如下函数进行文件名的匹配：

```
My_filename :=      My_Data_Module.My_Curr_Path.Curr_Nd_Path + '\' +
                          //合同管理文件夹
                          HTGLWJJ + '\' +
                          //合同文件夹
                          HTWJJ    + '\' +
                         //合同文件名前缀
                         HTQZ_WJM +
                         '??????????' ;
```

调入的核心代码详见附录B—调入合同函数。

4.4.2 内存流设计算法

1.本内存流的设计采用流线 型结构设计，每个内存流包含位置Position和大小Size属性，位置等于上一个数据流的位置加上上一个数据流的长度Length，大小等于所有数据流长度的总和。下面用合同首页和资金计划表格举例演示内存流位置和大小的计算方法：

```
My_memo_1.Position := My_Mem_Position;
My_memo_1.WriteBuffer(FileInfo,SizeOf(FileInfo));
My_memo_1.WriteBuffer(HTGL,SizeOf(HTGL));
//导入内存流--合同首页
SYBG_NCL( );
k := 0;
HTGL.PS[k] := SizeOf(FileInfo)+ SizeOf(HTGL);
HTGL.LEN[k] := My_memo.Size;
My_memo_1.WriteBuffer(My_memo.Memory^,HTGL.LEN[k]);
//-----------------------------------------------
//导入内存流--资金计划
ZJJHBG_NCL( );
k := k+1;
HTGL.PS[k] := HTGL.PS[k-1]+HTGL.LEN[k-1];
HTGL.LEN[k] := My_memo.Size;
My_memo_1.WriteBuffer(My_memo.Memory^,HTGL.LEN[k]);
```

2.每个文件都有其相对应的结构体作为数据的存储空间，与结构函数相对应的是析构函数，当数据存储完成后用析构函数来释放结构体。几个典型的文件结构见附录B。每个内存流的建立都要经过内存流清空My_memo_1.Clear，内存流位置初始化My_memo_1.Position := 0，内存流空间分配，其中包括头文件空间分配和主数据流空间分配My_memo_1.WriteBuffer(Struct, Size)，写入内存流My_memo_1.WriteBuffer。其主要代码如下：

```
My_memo_1.Clear;
My_memo_1.Position := 0;
My_memo_1.WriteBuffer(FileInfo.Ver , SizeOf( FileInfo.Ver ));
My_memo_1.WriteBuffer(HTML , SizeOf( HTML ));
//导入内存流
DRNCL( SizeOf( HTML )+ SizeOf( FileInfo.Ver )    );
```

```
FileInfo.Ver.SX1 :=    FileInfo.FileLength;
My_memo_1.Position := 0;
My_memo_1.WriteBuffer(FileInfo.Ver , SizeOf( FileInfo.Ver ));
```

4.5　系统分析结果测试

本系统设计在过程中一直使用Delphi 09进行系统的设计和调试，最终各方面测试结果如下。

界面显示效果：系统界面显示效果良好，没有出现乱码、颜色搭配不协调；页面内容充实、显示正确；表格大小适中随数据的录入自动调整，数据录入自动换行。点击按钮时，按钮发生颜色变化并且抖动，给用户良好的点击感觉。

用户验证：不同权限用户具有严格的验证，并且通过验证进入相应的界面；用户的账号、密码输入错误会有相应的提示；

功能实现：单位设置，对签约单位基本属性的设置，其中包含单位编码、单位名称、单位机构代码、单位账号等等；起草合同，起草合同的基本属性，资金计划，合同执行情况，合同文本，合同执行中的来往单位及备注；修改合同，对已经起草的合同进行修改，更新；合同类别设置：增加删除合同类别，可导入导出 EXCEL 文件；权限设置，增加删除操作人员并设置操作人员的操作权限；修改密码，修改当前操作员登录密码及基本信息；端口设置，网络连接设置可以实现异地办公；设置时间，系统时间设置；升级，可以导入系统升级文件；提交流程，实现了合同逐级审核审批，提交和回退。各模块的流程清晰，符合用户操作习惯，界面友好给用户良好的使用感受。

第 5 章　结论及展望

5.1　本文总结

本次毕业设计首先通过学院图书馆、网络资源等各种途径对数据仓库相关知识进行了复习，了解了信息管理系统的基本知识，掌握了合同管理系统的基本的方法。然后参考企业合同管理方面的文章，在网络上和现实中对合同管理现状进行了充分的需求分析和功能分析。从合同管理和系统设置两个大方面设计，各模块的实现都需自定义的内存流的支持。使本系统的功能基本能够实现，形成一个完整的系统，完成了毕业设计的任务。

其中，合同管理模块下：

单位设置：对签约单位基本属性的设置，其中包含单位编码、单位名称、单位机构代码、单位账号等，遍历起草合同时增加的来往单位基本信息，并对其进行修改保存。

增加合同：起草合同的首页基本属性，资金计划，合同执行情况，合同文本，合同执行来往单位及备注。

修改合同：对已经起草的合同进行修改，更新，从目录遍历已经起草完成的合同对合同状态进行逐级提交，签约-执行-完成。

合同类别设置：增删改查合同类别，方便合同起草修改时对类别的调入，可导入导出 EXCEL 文件。

系统设置模块下：

权限设置：增加删除操作人员并设置操作人员的操作权限；

修改密码：修改当前操作员登录密码及基本信息；

端口设置：网络连接设置，端口、ip 地址的修改；

设置时间：系统当前时间设置；

升级：导入系统升级文件。

由于因时间和能力方面的关系，本次毕业设计合同管理系统的研究仍有很多不足之处。如需求分析的不全面使无法实现合同管理系统所有功能需求。

通过本次毕业设计，不但学到了很多书本上课堂上无法学到的东西，而且在应用知识解决实际问题的方面有了很大的提高。在学习思维和动手能力上面对自己都有很大的提高。回顾系统的开发与论文撰写过程，有几多收获，也有诸多不尽人意之处，更需进一步的努力和开拓。

5.2　展望

本系统设计以企业的经济效益为目标，能够为企业管理人员和员工提供简单易用、功能强大并高度灵活的应用工具，激励他们的积极性，这些改进为企业带来更高的效率为企业创造利益。同时，通过对人流、物流、资金流的科学管理和

有效控制，提高员工的工作效率，降低各种经营成本，从而获取持久的利润。

本系统完成仅仅是合同管理系统的初步雏形，由于缺少实际企业相应合同的录取和非开发人员的测试，本系统在实际使用过程中必定存在诸多不便，但是在合同管理系统的使用过程中我们将通过用户对其功能的反馈，不断地进行相应的改善，使用系统的升级功能导入更新改进过的系统，不断的完善系统，使系统的界面及功能都更加的人性化。

现在以及将来一定会是一个信息化、自动化管理的高效率的社会，所以MIS必将的到一个长远普及的发展。

参考文献

[1].张岭,宋坤,梁冰. Delphi 程序开发典例宝典. 2006.7.
[2].罗斌. Delphi for.NET 编程实例精粹. 2006.5.
[3].明日科技,梁冰,宋坤. Delphi 范例完全自学手册. 2004.6.
[4].吕宗智,王世攀,王颢等. Delphi 实用技术精辟.2000.3.
[5].赵万军. Delphi 软件项目开发实例.2004.11.
[6].周国宏,罗述谦,罗起. Delphi 程序设计. 2006.4.
[7].周兴华. Delphi.NET 程序设计. 2004.
[8].郭振斌,黄业清. Delphi 高级页面特效制作百例.2006.6.
[9].江孝宜. 信息系统开发. 电子工业出版社,2006.7.
[10].中国交通建设监理协会组织. 合同管理. 人民交通出版社, 2013.5.
[11].孙红新. 公司合同管理. 法律出版社, 2007.10.1.
[12].陈燕青. 管理信息系统在公司中的应用 [J]. 电子制作, 2014.1.6.
[13].匡南. 浅谈中小企业管理信息系统现状及对策 [J]. 商场现代化, 2014.19.
[14]霍特尔曼. Head First C#.Southeast University Press,2009.5.1-131.
[15]沃尔斯. Spring in Action .Renmin University of Post Press,2007.

致 谢

经过一学期的努力,我的毕业设计终于完成了。在做毕业设计的这段时间里，我遇到了许多的问题，许多时候都不知道怎么进行下一步工作，每次在这时候，都会有老师、同学、朋友给我耐心的指导，细心的帮我解决问题。在此，我向他们表示最真诚的感谢。

首先，我要感谢王威老师耐心的教导和指导我们。从最初的开题报告和任务书，王老师都很认真负责的为我们每一个人的文档仔细的标注，告诉我们自己的问题，同时鼓励我们自己好好创新，如果有自己想做的毕业设计题目可以和他说，已经确定题目的同学，王老师鼓励他们好好做，并且每周都会检查一次，来告诉我们的缺点和不足在哪里，以便我们及时调整。同时，如果系里面开会有什么通知，他都会第一时间发邮件告诉我们，好让我们知道什么时候会第一次检查、中期检查和终期检查，来合理分配自己的时间。并且，王老师会经常到毕设教室来指导我们，为我们解答所遇到问题，给了我们很大的帮助。

其次，我要感谢我们网络工程系的所有老师。在第一次检查、中期检查中，他们给我提出了很多的宝贵意见和建议，还经常督促我们抓紧时间做毕业设计。我要感谢帮助我的同学们，他们给了我很大的启发与帮助，帮助我解决了诸多在编写程序中遇到的问题。同时，也很感谢这次毕业设计，它让我懂得了学习的乐趣，当自己经过辛苦努力钻研出来一个劳动成果的时候，那种心情是无法比拟的，在这个过程中我也学到了许多从前没有学过的知识与经验，让我收获很多。

最后感谢信息学院给大家提供的毕业设计实验室。向信息学院所有的老师表示深深的谢意。

附录 A
英文原文

Developing applications with Delphi

Borland Delphi is an object-oriented, visual programming environment to develop32-bit applications for deployment on Windows and Linux. Using Delphi, you can create highly efficient applications with a minimum of manual coding.

Delphi provides a suite of Rapid Application Development (RAD) design tools, including programming wizards and application and form templates, and supports object-oriented programming with a comprehensive class library that includes:

The Visual Component Library (VCL), which includes objects that encapsulate the Windows API as well as other useful programming techniques (Windows).

The Borland Component Library for Cross-Platform (CLX), which includes objects that encapsulate the Qt library (Windows or Linux).

This chapter briefly describes the Delphi development environment and how it fits into the development life cycle. The rest of this manual provides technical details on developing general-purpose, database, Internet and Intranet applications, creating ActiveX and COM controls, and writing your own components.

Integrated development environment

When you start Delphi, you are immediately placed within the integrated development environment, also called the IDE. This IDE provides all the tools you need to design, develop, test, debug, and deploy applications, allowing rapid prototyping and a shorter development time.

The IDE includes all the tools necessary to start designing applications, such as the:

Form Designer, or form, a blank window on which to design the user interface (UI) for your application.

Component palette for displaying visual and non visual components you can use to design your user interface.

Object Inspector for examining and changing an object's properties and events.

Object Tree View for displaying and changing a components' logical relationships.

Code editor for writing and editing the underlying program logic.

Project Manager for managing the files that make up one or more projects.

Integrated debugger for finding and fixing errors in your code.

Many other tools such as property editors to change the values for an object's property.

Command-line tools including compilers, linkers, and other utilities.

Extensive class libraries with many reusable objects. Many of the objects provided in the class library are accessible in the IDE from the Component palette. By convention, the names of objects in the class library begin with a T, such as TStatusBar. Names of objects that begin with a Q are based on the Qt library and are used for cross-platform applications.

Some tools may not be included in all editions of the product.

A more complete overview of the development environment is presented in the Quick Start manual included with the product. In addition, the online Help system provides help on all menus, dialog boxes, and windows.

Designing applications

You can design any kind of 32-bit application—from general-purpose utilities to sophisticate data access programs or distributed applications.

As you visually design the user interface for your application, the Form Designer generates the underlying Delphi code to support the application. As you select and modify the properties of components and forms, the results of those changes appear automatically in the source code, and vice versa. You can modify the source files directly with any text editor, including the built-in Code editor. The changes you make are immediately reflected in the visual environment.

You can create your own components using the Delphi language. Most of the components provided are written in Delphi. You can add components that you write to the Component palette and customize the palette for your use by including new tabs if needed.

You can also design applications that run on both Linux and Windows by using CLX components. CLX contains a set of classes that, if used instead of those in the VCL, allows your program to port between Windows and Linux. Refer to Chapter 15, "Developing cross-platform applications" for details about cross-platform programming and the differences between the Windows and Linux environments. If you are using Kylix while developing cross-platform applications, Kylix also includes a Developer's Guide that is tailored for the Linux environment. You can refer to the

manual both in the Kylix online Help or the printed manual provided with the Kylix product.

Chapter 8, “Building applications, components, and libraries,” introduces support for different types of applications.

Creating projects

All application development revolves around projects. When you create an application in Delphi you are creating a project. A project is a collection of files that make up an application. Some of these files are created at design time. Others are generated automatically when you compile the project source code.

You can view the contents of a project in a project management tool called the Project Manager. The Project Manager lists, in a hierarchical view, the unit names, the forms contained in the unit (if there is one), and shows the paths to the files in the project. Although you can edit many of these files directly, it is often easier and more reliable to use the visual tools.

At the top of the project hierarchy is a group file. You can combine multiple projects into a project group. This allows you to open more than one project at a time in the Project Manager. Project groups let you organize and work on related projects, such as applications that function together or parts of a multi-tiered application. If you are only working on one project, you do not need a project group file to create an application.

Project files, which describe individual projects, files, and associated options, have a .dpr extension. Project files contain directions for building an application or shared object. When you add and remove files using the Project Manager, the project file is updated. You specify project options using a Project Options dialog which has tabs for various aspects of your project such as forms, application, and compiler. These project options are stored in the project file with the project.

Units and forms are the basic building blocks of an application. A project can share any existing form and unit file including those that reside outside the project directory tree. This includes custom procedures and functions that have been written as standalone routines.

If you add a shared file to a project, realize that the file is not copied into the current project directory; it remains in its current location. Adding the shared file to the current project registers the file name and path in the uses clause of the project file. Delphi automatically handles this as you add units to a project.

When you compile a project, it does not matter where the files that make up the project reside. The compiler treats shared files the same as those created by the project itself.

Editing code

The Code editor is a full-featured ASCII editor. If using the visual programming environment, a form is automatically displayed as part of a new project. You can start designing your application interface by placing objects on the form and modifying how they work in the Object Inspector. But other programming tasks, such as writing event handlers for objects, must be done by typing the code.

The contents of the form, all of its properties, its components, and their properties can be viewed and edited as text in the Code editor. You can adjust the generated code in the Code editor and add more components within the editor by typing code. As you type code into the editor, the compiler is constantly scanning for changes and updating the form with the new layout. You can then go back to the form, view and test the changes you made in the editor, and continue adjusting the form from there.

The code generation and property streaming systems are completely open to inspection. The source code for everything that is included in your final execute able file—all of the VCL objects, CLX objects, RTL sources, and project files—can be viewed and edited in the Code editor.

Compiling applications

When you have finished designing your application interface on the form and writing additional code so it does what you want, you can compile the project from the IDE or from the command line.

All projects have as a target a single distributable executable file. You can view or test your application at various stages of development by compiling, building, or running it:

When you compile, only units that have changed since the last compile are re compiled.

When you build, all units in the project are compiled, regardless of whether they have changed since the last compile. This technique is useful when you are unsure of exactly which files have or have not been changed, or when you simply want to ensure that all files are current and synchronized. It's also important to build when you've changed global compiler directives to ensure that all code compiles in the

proper state. You can also test the validity of your source code without attempting to compile the project.

When you run, you compile and then execute your application. If you modified the source code since the last compilation, the compiler recompiles those change dmodules and re links your application.

If you have grouped several projects together, you can compile or build all projects in a single project group at once. Choose Project Compile All Projects or Project Build All Projects with the project group selected in the Project Manager .Note To compile a CLX application on Linux, you need Kylix.

Debugging applications

With the integrated debugger, you can find and fix errors in your applications. The integrated debugger lets you control program execution, monitor variable values and items in data structures, and modify data values while debugging.

The integrated debugger can track down both runtime errors and logic errors. By running to specific program locations and viewing the variable values, the function son the call stack, and the program output, you can monitor how your program behaves and find the areas where it is not behaving as designed. The debugger is described in online Help.

You can also use exception handling to recognize, locate, and deal with errors. Exceptions are classes, like other classes in Delphi, except, by convention, they begin with an initial E rather than a T.

Deploying applications

Delphi includes add-on tools to help with application deployment. For example, Install Shield Express (not available in all editions) helps you to create an installation package for your application that includes all of the files needed for running adistributed application. Team Source software (not available in all editions) is also available for tracking application updates.

To deploy a CLX application on Linux, you need Kylix.

Note Not all editions have deployment capabilities.

中文译文

发展中的应用软件 Delphi

Delphi 是一个目标导向，可视化编程环境，来扩展 32 位的，使用 Delphi,您可以创建高效的应用程序的手工编码。

Delphi 提供了一套快速应用开发（RAD）设计工具，包括编程向导和模板的应用，并支持一个全面的类库，包括面向对象编程：可视化组件库（VCL），包括对象封装 Windows API 以及其他有用的编程技术（Windows）。跨平台 Borland 构件库（CLX），包括 (Windows or Linux)封装的 Qt 库（Windows 或 Linux）的对象。

本章简要介绍了 Delphi 开发环境和如何适应在开发生命周期。本手册提供的技术细节开发通用，数据库，Internet 和 Intranet 应用，创造 ActiveX 控件和 COM 控件，并编写自己的组件。

集成开发环境

当你开始 Delphi，你立即被放置在集成开发环境中也被称为 IDE。这个 IDE 提供了所有工具，你需要设计，开发，测试，调试，并部署应用程序，允许快速成型和发展的时间较短。

IDE 包括所有必要的工具来开始设计应用程序，如：

表单设计器，或形式，在其中一个空白窗口设计用户界面（UI）应用程序。

部件调色板显示视觉和各种你可以使用组件设计你的用户界面。

对象督察检查和改变对象的属性和事件。

对象树视图用于显示和改变一个组件的逻辑关系。

写作和编辑的基本程序逻辑代码编辑器。

管理，组成一个或多个项目文件，项目经理。

集成调试器查找和修正你代码中的错误。

许多其他工具，如财产编辑器来改变一个对象的值财产。

命令行工具，包括编译器，连接器，和其他公用事业。

广泛的类库有许多可重用的对象。许多的对象类库中提供了可以在 IDE 的组件面板。

通过公约，类库中的对象名称以 T 开头，如状态栏组件。对象与 Q 开始的名字是基于 Qt 库和用于跨平台的应用程序。一些工具可能不包括在产品的所有版本。一个比较完整的开发环境概述在快速入门手册包括与产品。此外，在线帮助系统提供帮助的所有的菜单，对话框和窗口。

设计应用程序

你可以设计任何一种从通用电力公司的 32 位应用精密的数据访问程序或分布式应用程序。你可以直观地设计应用程序的用户界面，窗体设计器下面的代码生成的应用程序支持。当你选择和修改成分和形态特征，这些变化的结果会自动出现在源代码中，反之亦然。你可以直接用任何文本编辑器修改源文件，包括内置的代码编辑器。您所做的更改会立即反映在视觉环境中。你可以使用 Delphi 语言创建自己的插件。大多数的组件提供了在 Delphi 编写。你可以添加组件到你写的为你的使用包括新标签如果组件和自定义调色板调色板需要。

你也可以设计，运行在 Linux 和 Windows 应用程序 CLX 组件。CLX 包含一组类，如果用来代替那些在 VCL，允许你的程序的 Windows 和 Linux 之间的端口。参见第 15 章， “开发跨平台的应用对跨平台的细节 Windows 和 Linux 编程环境之间的差异。如果你是用 Kylix 开发跨平台的应用程序，也包括 Kylix 一个开发者指南是专为 Linux 环境。你可以参考一下手动在 Kylix 在线帮助或说明书印刷有陶杯产品。第 8 章，“构建应用程序，组件，和图书馆，介绍了支持不同类型的应用。

创建项目

一切围绕项目的开发应用。当你创建一个 Delphi 应用程序创建一个项目。项目文件的收集，组成应用程序的，这些文件是在设计时创建。其他人自动生成编译项目时源代码。你可以在一个项目管理工具查看项目的内容称为项目经理。项目经理的名单，在一个分层的观点，单位名称，该包含在单元形式（如果有），并显示在文件路径项目。虽然你可以编辑这些文件直接，它往往是更容易和更使用视觉工具可靠。在项目层次结构的顶部是一组文件。你可以把多个项目到项目组。这允许你同时打开多个项目在项目经理。项目组让你组织和工作相关的项目，如应用程序或部分多层应用。如果你只有在一个项目上工作，你不需要一个项目组文件创建一个应用。项目文件，描述单个项目，文件，和相关的选项，有。朝鲜的延伸。项目文件包含了建设一个应用程序或方向共享对象。当您添加和使用项目经理删除文件，项目文件更新。您指定的项目使用项目选项对话框的选项有为你的项目等形式，应用各方面的标签，和编译器。这些项目的选项存储在项目文件。单位和窗体是应用程序的基本构建块。一个项目可以分享任何现有的窗体和单元文件包括那些居住在项目目录树。这包括自定义的过程和函数，编写了作为独立的程序。如果你添加一个共享文件的一个项目，实现不复制该文件到当前项目目录；它仍然在其当前位置。添加共享文件目前项目注册文件的文件名和路径的使用条款中的项目文件。Delphi 自动处理为你增加单位工程。当你编译一个项目，不要紧，文件组成项目居住。编译器将共享文件为这些项目所创造的一样本身。

编辑代码

代码编辑器是一个功能齐全的 ASCII 编辑器。如果使用可视化编程环境，一种是自动显示为新计划的一部分。你可以开始通过将对象的形式和修改设计你的应用程序接口他们如何在对象督察工作。但其他的编程任务，如写作对象的事件处理程序，必须输入代码完成。表格的内容，它的所有属性，它的成分，和他们的属性可以查看和编辑在代码编辑器中的文本。你可以调整生成的代码，在代码编辑器中添加更多的组件内的编辑器输入代码。作为你的类型的代码到编辑器，编译器是不断扫描随着新的布局更新形式的变化。然后你可以回到形式，查看和测试编辑器中所做的更改，并不断调整形式从那里。代码生成和性能的流媒体系统是完全开放的检查。都是包含在最终的可执行文件的源代码文件所有的 VCL 对象，CLX 对象，RTL 源，和项目文件可以在代码编辑器中查看和编辑。

编译应用程序

当你已经完成了你想要的形式上的应用程序接口设计和写额外的代码，你可以编译项目 IDE 或命令行程序。所有的项目都作为一个对象由一个单一分配可执行文件。你可以在编译中查看或测试你的应用程序在不同发展阶段的构筑，或运行它：

当你编译的时候，从最后的编译单一单元已经被改变需要再编译。

当你建立的时候，项目中的所有单元都被编译了，不管他们是否已经改变了最后的编译。当你不确定哪些文件改变或没有改变，或当你只是要确保所有的文件都是电流和同步的时候这种技术是有用的。它的建立也很重要，当你改变了全部的编译器指令来确保所有的代码编写适当的状态。你也可以测试你的源代码的有效性没有尝试编译项目。

当你运行的时候，你编译并执行你的应用程序。如果你修改自上次编译的源代码，编译器会重新编译那些已经改变的模块，重新连接你的应用程序。

如果你已经把几个项目组在一起，你可以一次性编译或建立所有项目在一个项目组中。编译所有项目或选择项目工程建设项目中所有项目经理选择项目组。

Note 编译一个 CLX 应用程序在 Linux，你需要 Kylix。

调试应用程序

通过集成的调试器，你可以找到和解决你的应用程序错误。集成调试器可以控制程序的执行，监视数据结构的项目中的变量值，并在调试中修改数据值。

集成调试器可以跟踪运行时间错误和逻辑错误。通过运行特定的程序位置和查看变量的值，函数在“调用堆栈”，和程序的输出，您可以监视您的程序是怎样运行的和找到设计中没被运行的区域。调试器的使用方法可以在在线帮助中查找到。

你也可以使用异常处理来识别，定位，和处理错误。异常类，像其他类在 Delphi 中，除了特殊的类，一般按照惯例来设计，他们都是用 E 开头而不是用 T 开头。

部署应用程序

Delphi 包含用来帮助应用程序部署的附加工具。例如，InstallShield Express（所有版本中不可用）可以帮助你创建一个您的应用程序的安装包，包括所有运行所需的文件包分布式应用。teamsource 软件（所有版本中不可用）也可用于跟踪应用程序更新。在 Linux 系统下部署一个 CLX 应用程序，你需要 Kylix。

Note 不是所有的版本都有部署能力。

附录 B
几个典型的文件结构

1. //文件头信息数据结构

```
FileInfoStruct = record
        Ver :    Word ; //版本号
        SX   :   int16;//属性
        FileLengthHead : Int32;//文件头长度
        FileLength : Int64;//文件大小
        RecordCount : Int64;//文件大小
        SystemTime : SystemTimeStruct;//日期
        Explain : array[0..100] of Char;//备注
    end;
PFileInfoStruct = ^FileInfoStruct ;
```

2. //合同管理数据结构

```
HTGL_Struct = record
        count : Int32;
        PS : array [0..10] of Int32;
        LEN : array [0..10] of Int32;
    end;
```

3. //合同首页数据结构

```
        FL_HTSYJG_Struct=record
            HTSX:Int8 ;                  //合同属性
            HTBH:Int64 ;       //合同编号
            HTMC:array [0..30 ] of Char ;       //合同名称
            HTLB:array [0..20 ] of Char ;       //合同类别
            QCBM:array [0..50 ] of Char ;     //起草部门
            HTQSSJ : JianYueTimeStruct ;           //合同起始时间
            HTZZSJ : JianYueTimeStruct ;           //合同终止时间
            QCBMYJ:array [0..100 ] of Char ;   //起草部门意见
            QCBMYJ_FZR:array [0..7 ] of Char ;     //起草部门负责人
            QCBMYJ_FZR_SJ : JianYueTimeStruct ;   //起草部门负责人签字时
间
            QCBMYJ_FZR_TY : int8;     //起草部门true同意、false拟同意
            QCBMYJ_FHR:array [0..7 ] of Char ;     //起草部门复核人
```

QCBMYJ_FHR_SJ : JianYueTimeStruct ; //起草部门复核人签字时间

QCBMYJ_FHR_TY : int8; //起草部门true同意、false拟同意

QCBMYJ_JBR:array [0..7] of Char ; //起草部门经办人

QCBMYJ_JBR_SJ : JianYueTimeStruct ; //起草部门经办人签字时间

QCBMYJ_JBR_TY : int8; //起草部门true同意、false拟同意

QCBMYJ_LXDH:array [0..20] of Char ; //联系电话

CWBMYJ :array [0..100] of Char ; //财务部门意见

CWBMYJ_FZR:array [0..7] of Char ; //财务部门负责人

CWBMYJ_FZR_SJ : JianYueTimeStruct ; //财务部门负责人签字时间

CWBMYJ_FZR_TY : int8; //财务部门true同意、false拟同意

CWBMYJ_FHR:array [0..7] of Char ; //财务部门复核人

CWBMYJ_FHR_SJ : JianYueTimeStruct ; //财务部门复核人签字时间

CWBMYJ_FHR_TY : int8; //财务部门true同意、false拟同意

CWBMYJ_JBR:array [0..7] of Char ; //财务部门经办人

CWBMYJ_JBR_SJ : JianYueTimeStruct ; //财务部门经办人签字时间

CWBMYJ_JBR_TY : int8; //财务部门true同意、false拟同意

CWBMYJ_LXDH:array [0..20] of Char ; //联系电话

LSBMYJ :array [0..100] of Char ; //法律部门意见

LSBMYJ_FZR:array [0..7] of Char ; //法律部门负责人

LSBMYJ_FZR_SJ : JianYueTimeStruct ; //法律部门负责人签字时间

LSBMYJ_FZR_TY : int8; //法律部门true同意、false拟同意

LSBMYJ_FHR:array [0..7] of Char ; //法律部门复核人

LSBMYJ_FHR_SJ : JianYueTimeStruct ; //法律部门复核人签字时间

LSBMYJ_FHR_TY : int8; //法律部门true同意、false拟同意

LSBMYJ_JBR:array [0..7] of Char ; //法律部门经办人

LSBMYJ_JBR_SJ : JianYueTimeStruct ; //法律部门经办人签字时间

```
            LSBMYJ_JBR_TY : int8;     //法律部门true同意、false拟同意
            LSBMYJ_LXDH:array [0..20 ] of Char ;     //联系电话
            LDQZ :array [0..100 ] of Char ;          //领导签字
            LDQZYJ_FZR:array [0..7 ] of Char ;     //领导签字负责人
            LDQZYJ_FZR_SJ : JianYueTimeStruct ;   //领导签字负责人签字时
间
            LDQZYJ_FZR_TY : int8;     //领导签字true同意、false拟同意
            LDQZYJ_FHR:array [0..7 ] of Char ;     //领导签字复核人
            LDQZYJ_FHR_SJ : JianYueTimeStruct ;   //领导签字复核人签字时
间
            LDQZYJ_FHR_TY : int8;     //领导签字true同意、false拟同意
            LDQZYJ_JBR:array [0..7 ] of Char ;     //领导签字经办人
            LDQZYJ_JBR_SJ : JianYueTimeStruct ;   //领导签字经办人签字时
间
            LDQZYJ_JBR_TY : int8;     //领导签字true同意、false拟同意
            LDQZYJ_LXDH:array [0..20 ] of Char ;     //联系电话
            BZ:array [0..80 ] of Char ;              //备注
        end;
         P_FL_HTSYJG_Struct = ^FL_HTSYJG_Struct;
    //调入合同函数
    procedure THTGL_Form.DRHT();
    var
      i , k: Integer;
    begin
        //读入当前合同状态
        My_memo_ML.Position:=
        SizeOf(FileInfo) + SizeOf(HTML)*(Ej_HTML.CurRow-1);
        My_memo_ML.ReadBuffer(HTML,SizeOf(HTML));
        //提交状态显示
        Tijiao_Zhuangtai();
        //---------------------------------------------------------------------
        BZ1 := False;
        //内存流清空
        My_memo.Clear;
```

```
        My_memo.Position := 0;
        Ej_HTML.Editing := False;
        old_ML := Ej_HTML.CurRow ;
        My_File_Name                                                    :=
My_StrFront(Ej_HTML.Cells[1,Ej_HTML.CurRow].Text,':');
        My_File_Name := My_Data_Module.My_Curr_Path.Curr_Nd_Path   + '\' +
                                    //合同管理文件夹
                                    HTGLWJJ + '\' +
                                    //合同文件夹
                                    HTWJJ   + '\' +
                                    //合同文件名前缀
                                    HTQZ_WJM+
                                    My_File_Name ;
        My_memo_1.Clear;
        if     My_Read_Sever_File(
                                My_File_Name,//文件名
                                My_memo_1,//文件内存流
                                0,//文件从 0 偏移量  -1:不偏移
                                -1,//文件长度  -1:取全部文件)   > 0
        then
        begin
            My_memo_1.Position := 0;
            My_memo_1.ReadBuffer(FileInfo,SizeOf(FileInfo));
            My_memo_1.ReadBuffer(HTGL,SizeOf(HTGL));
            //内存到合同首页表
            k := 0;
            My_memo.Clear;
            My_memo.Size :=HTGL.LEN[k];
            My_memo.Position :=0;
            My_memo_1.Position := HTGL.PS[k];
            My_memo_1.ReadBuffer(My_memo.Memory^ ,HTGL.LEN[k]);
            NC_HTSY( ) ;
            //------------------------------
            //内存到资金计划
```

```
        k := k + 1;
        My_memo.Clear;
        My_memo.Size :=HTGL.LEN[k];
        My_memo.Position :=0;
        My_memo_1.Position := HTGL.PS[k];
        My_memo_1.ReadBuffer(My_memo.Memory^ ,HTGL.LEN[k]);
        NC_HTZJJH ( );
end;
end;
```