毕业设计（论文）材料之二（1）

# 安徽工程大学本科

# 毕业设计（论文）

专　　业：物联网工程

题　　目：汽车部件公司仓储管理

作 者 姓 名：张中华

导师及职称：王勇（副教授）

导师所在单位：计算机与信息学院

**2017** 年　月　日

# 安徽工程大学

# 本科毕业设计（论文）任务书

**2017** 届 计算机与信息 学院

物联网工程 专业

学生姓名：张中华

## Ⅰ 毕业设计（论文）题目

**中文：汽车部件公司仓储管理**

**英文：Auto parts company warehouse management**

## Ⅱ 原始资料

[1] Christian Nagel，Jay Glynn，Morgan Skinner.C#高级编程（第 9 版），2016-08

[2] Ben-Gan.Microsoft SQL Server 2008 技术内幕 T-SQL 语言基础.2009.10

[3] 王珊,萨师煊.数据库系统概论（第四版）.高等教育出版社，2006.5

[4] 王国辉,王毅.数据库系统开发案例精选[M].人民邮电出版社,2006

[5] Baron Scbwartz.高性能 MySQL（第 3 版）[M].电子工业出版社，2013

[6] （美）加洛韦等著.ASP.NET MVC 5 高级编程(第 5 版)[M],2015-02

[7] 构建之法 现代软件工程（第二版）[M],2015-07

[8] 王国辉,王毅.数据库系统开发案例精选[M].人民邮电出版社,2006

[9] 软件工程：实践者的研究方法（原书第 8 版）,2016-11

[10] 程成，陈霞.软件工程[M].机械工业出版社，2003

# Ⅲ 毕业设计（论文）任务内容

## 1、本课题的目的与意义：

针对汽车部件公司的实际需求，调研汽车部件种类，型号等基本信息，了解仓储管理的实际工作流程。基于软件工程的设计思想设计与实现汽车部件公司的仓储管理系统。借助于该管理信息系统课题，使学生能掌握信息管理信息系统开发的一般方法，能利用开发工具开发较简单的管理信息系统。主要内容包括：（1）数据库设计；（2）基础信息设置；（3）入库管理；（4）销售管理；（5）库存管理；（6）查询及报表。

## 2、本课题任务的主要内容：

（1）熟练掌握一门软件开发工具及数据管理系统如 Sql Server。

（2）搜集和整理资料，阅读一定量中外文文献，掌握汽车部件仓储管理系统的分析与设计方法。

（3）调研和分析，确定系统需求；按软件工程的方法设计系统。。

（4）开发一套能满足实际应用要求的汽车部件公司的仓储管理系统，主要功能有：数据库设计、基础信息设置、入库管理、销售管理、库存管理、查询与报表等。

## 3、提交的成果：

（1）毕业设计（论文）正文；

（2）提交系统原型；

（3）至少一篇引用的外文文献及其译文；

（4）附不少于 10 篇主要参考文献的题录及摘要。

指导教师（签字）_____

教研室主任（签字）_____

批　准　日　期 _____

接受任务书日期 _____

完　成　日　期 _____

接受任务书学生（签字）_____

# 汽车部件仓储管理系统

## 摘　　要

随着信息技术的迅速发展,信息化管理已经成为企业应对挑战,参与市场竞争、降低管理成本的一种重要方式。汽车部件仓储管理系统帮助企业进行汽车部件信息管理,整合部件信息,处理部件出库、入库、盘点等复杂业务,帮助企业在对业务的管理过程中提取重要数据,实现长期规划,提高企业管理者的管理水平和业务能力,为企业发展决策提供科学依据,实现高效的管理。

该文首先分析了汽车部件信息管理的实际需求，具体描述了整个系统的开发过程,给出了汽车部件信息管理系统的关键功能的实现过程。采用 Microsoft Visual Studio 和 Sql Server 作为软件开发工具，完成系统的关键功能，并采用单元测试方法对关键模块进行测试。系统的主要功能包括：汽车部件信息的管理，供应商和需求商的管理，出库和入库等。

所实现的汽车部件仓储管理系统原型能有效提高仓库的存储效率,减少仓库管理漏洞,具有对汽车部件信息、供应商信息、所涉及员工信息进行管理及维护的功能,也具有对每一次汽车部件的出库、入库进行记录的功能。

**关键词**：信息管理；仓储管理；汽车部件；Microsoft Visual Studio

# Management System for Auto Parts Warehouse

## Abstract

With the rapid development of information technology, information management has become a challenge, take part in market competition, an important method of reducing management costs. Auto parts auto parts warehouse management system helps enterprises information management and integration of parts information, parts, warehousing, inventory and other complex business to help businesses extract important data in the management of business processes, long term planning and improve enterprise's management and operational capacity, provide the scientific basis for decision-making of enterprise development, to achieve efficient management.

In order to achieve effective automobile parts warehousing and information management, this system introduces the demand for auto parts information management, description of the system development process, analyzes the function of information management system for automobile parts, using Microsoft Visual Studio and Sql Server tools to the realization of the function of the code, and detailed records of each test case information. Detailed design and system database, establishing the e-r model of process management modules, logic and database logic to the data table structure was designed. Finally, the overall design of the system, each function is described in detail.

Auto parts warehouse management system is on auto parts information is an important tool for effective management. Our goal is to improve the efficiency of warehouse storage, reduce warehouse management loopholes for automotive parts information, supplier information, management and maintenance functions of the involved employee information, also for every auto parts library functions for recording, storage.

**Key words**: Information management ;system ;auto parts; Microsoft Visual Studio

# 目录

# 插图清单

# 表格清单

# 引言

随着计算机技术的飞速发展，许多企业事业单位的管理都实现了管理自动化，这种自动化管理方式不仅管理简单，而且效率非常高。为了能够高效地管理汽车部件的仓储管理信息，汽车部件经营者提出使用计算机进行汽车部件信息的管理，使汽车部件管理科学化，最大限度地减少信息损失，提高汽车部件生产的利益。

使用计算机管理汽车部件相对人工记录，有很多的优点。首先，用计算机进行计算时速度快，可信度高。而且查询时不必要逐个查找，只需要输入相关信息就可快速得到结果。然后，汽车部件信息存储在计算机，可以做到数据的永久保存，安全可靠。最重要的是，汽车部件数据存储在计算机中，由于计算机存储容量非常大，所以清单的内容在输入电脑后，对数据的操作是非常方便的，而且避免了频繁的使用清单。

汽车部件仓储管理已经渐渐的走向稳定发展的趋势，更加具有企业化的概念，在体制上，汽车部件仓储管理已经开始慢慢的健全它的体制，对公司人员进行培训，将业务进行熟练化，这样大大的提高了汽车部件管理的发展，并且使它的地位明显的上升，也进一步的满足了大家的需求。为了更好的发展，企业渐渐的从整体中分离开来建立自己的发展模块，不断的寻求发展模式，扩大自己的经营模式。

本系统使用Microsoft Visual Studio和SQL Server等工具，完成了系统对汽车部件的信息进行管理，不但可以使用工程的、规范的管理过程，而且可以有效的提高了工作人员的工作效率，直观的、科学的管理汽车部件信息，进而完成公司的业务，这对汽车部件信息的管理的发展及信息化的管理具有极其重要的意义。

# 第 1 章 绪 论

## 1.1 背景

汽车部件仓储管理系统的目的是为企业提供一个计算机化的管理平台，实践企业内部科学有效的管理，促进企业管理信息化，规范化，将能使管理人员从繁琐的杂务工作中解脱出来，真正从事管理工作。

目前汽车部件生产企业对汽车部件仓储的管理还是手工进行，随着汽车行业的迅速发展，手工管理汽车部件的种种弊端暴露无疑，给销售企业的发展带来了不必要的麻烦。为了规范企业内部管理，提高企业业务管理水平，更好的为客户服务，应采用计算机来管理汽车部件的进销存业务。

汽车部件仓储管理已经渐渐的走向稳定发展的趋势，更加具有企业化的概念，在体制上，汽车部件仓储管理已经开始慢慢的健全它的体制，对公司人员进行培训，将业务进行熟练化，这样大大的提高了汽车部件管理在世界中的发展，并且使它的地位明显的上升，也进一步的满足了大家的需求。为了更好的发展，企业渐渐的从整体中分离开来建立自己的发展模块，不断的寻求发展模式，扩大自己的经营模式。

## 1.2 系统开发的意义

随着计算机技术的快速发展，许多企业事业单位的管理都实现了办公自动化，这种自动化管理方式不仅管理简单，而且效率非常高。为了能够高效而且有效地管理汽车部件的管理信息，汽车部件经营者提出使用计算机进行汽车部件信息的管理，使汽车部件管理科学化，最大限度地减少汽车部件管理信息的损失，提高汽车部件生产的利益。

使用计算机管理汽车部件相对人工记录，有很多的优点。首先，用计算机进行速度快。而且查询时不必要逐个查找，只需要输入相关信息就可快速得到结果。然后，汽车部件信息存储在计算机，可以作到数据的永久保存，安全可靠。最重要的是，汽车部件数据存储在计算机中，由于计算机存储容量非常大，所以清单的内容在输入电脑后，对数据的操作是非常方便的，而且避免了频繁的使用清单。

## 1.3 研究现状和发展趋势

### 1.3.1 研究现状

现在我国的企业特别是汽车部件生产企业的管理水平还停留在半自动（由电脑处理一部分数据，由人工处理一部分数据）的基础上，这样的机制已经不能适应时代的发展，因为它浪费了许多人力和物力，在信息时代这种传统的管理方式必然会被以计算机为基础的信息管理所取代。软件作为一项有力的工具，只能当此种工具，与我们的实践相结合起来的时候，才具有重大的社会价值及使用价值。因此根据企业目前实际的汽车部件管理系统情况开发一套汽车部件管理系统是十分有必要的。

### 1.3.2 发展趋势

汽车部件仓储管理已经渐渐的走向稳定发展的趋势，更加具有企业化的概念，在体制上，汽车部件仓储管理已经开始慢慢的健全它的体制，对公司人员进行培训，将业务进行熟练化，这样大大的提高了汽车部件管理在世界中的发展，并且使它的地位明显的上升，也进一步的满足了大家的需求。为了更好的发展，企业渐渐的从整体中分离开来建立自己的发展模块，不断的寻求发展模式，扩大自己的经营模式。

通过计算机进行汽车部件的入库和出库数量的统计，使管理者不必再为统计数量而感到烦恼，可以利用节省出来的时间全身心的投入到其他事情当中，也对管理制度进行

了优化和改良，集中对零部件进行统计和分配，这样不仅减少我们使用的资源，也大大的降低了我们的劳动成本，节省了财力，使管理人员更加专心的从事管理工作，使管理制度更加合理化和规范化。

# 第 2 章 系统开发工具及相关技术

本章将对本次汽车部件信息管理系统的开发平台进行简要的介绍，同时介绍在开发过程中采用的一些 Ajax 技术。

## 2.1 开发工具简介

本系统主要采用的是学校中通用的软件，操作系统是 Windows，主要 Web 端和后端的开发都是在 Microsoft Visual Studio 中操作实现，数据库采用的是 Sql Server 数据库。下面将对两个软件简要介绍：

### 2.1.1 Microsoft Visual Studio 简介

Microsoft Visual Studio（简称 VS）是微软公司的开发工具包系列产品。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，包括 Microsoft Windows、Windows Phone、Windows CE、.NET Framework、.NET Compact Framework 和 Microsoft Silverlight。

而 Visual Studio .NET 是用于快速生成企业级 ASP.NET Web 应用程序和高性能桌面应用程序的工具。Visual Studio 包含基于组件的开发工具（如 Visual C#、Visual J#、Visual Basic 和 Visual C++），以及许多用于简化基于小组的解决方案的设计、开发和部署的其他技术。

### 2.1.2 Sql Server 简介

SQL Server 一开始并不是微软自己研发的产品，而是当时为了要和 IBM 竞争时，与 Sybase 合作所产生的，其最早的发展者是 Sybase[1]，同时微软也和 Sybase 合作过 SQL Server 4.2 版本的研发，微软亦将 SQL Server 4.2 移植到 Windows NT（当时为 3.1 版），在与 Sybase 终止合作关系后，自力开发出 SQL Server 6.0 版，往后的 SQL Server 即均由微软自行研发。

Sql Server 是一个关系数据库管理系统。具有易用性、适合分布式组织的可伸缩性、用于决策支持的数据仓库功能、与许多其他服务器软件紧密关联的集成性、良好的性价比等。为数据管理与分析带来了灵活性，允许单位在快速变化的环境中从容响应，从而获得竞争优势。

维基百科给的解释是这样的，Microsoft SQL Server 是由美国微软公司所推出的关系数据库解决方案，最新的版本是 SQL Server 2016，已经在 2016 年 6 月 1 日发布。 数据库的内置语言原本是采用美国标准局（ANSI）和国际标准组织（ISO）所定义的 SQL 语言，但是微软公司对它进行了部分扩充而成为作业用 SQL（Transact-SQL）。 几个初始版本适用于中小企业的数据库管理，但是近年来它的应用范围有所扩展，已经触及到大型、跨国企业的数据库管理。

## 2.2 .net 平台介绍

.NET 就是微软用来实现 XML，Web Services，SOA（面向服务的体系结构 service-oriented architecture）和敏捷性的技术。对技术人员，想真正了解什么是.NET，必须先了解.NET 技术出现的原因和它想解决的问题，必须先了解为什么他们需要 XML,Web Services 和 SOA。技术人员一般将微软看成一个平台厂商。微软搭建技术平台，而技术人员在这个技术平台之上创建应用系统。从这个角度，.NET 也可以如下来

定义：.NET 是微软的新一代技术平台，为敏捷商务构建互联互通的应用系统，这些系统是基于标准的，联通的，适应变化的，稳定的和高性能的。从技术的角度，一个.NET 应用是一个运行于.NET Framework 之上的应用程序。（更精确的说，一个.NET 应用是一个使用.NET Framework 类库来编写，并运行于公共语言运行时 Common Language Runtime 之上的应用程序。）如果一个应用程序跟.NET Framework 无关，它就不能叫做.NET 程序。比如，仅仅使用了 XML 并不就是.NET 应用，仅仅使用 SOAP SDK 调用一个 Web Service 也不是.NET 应用。.NET 是基于 Windows 操作系统运行的操作平台，应用于互联网的分布式。

## 2.3 相关技术

### 2.3.1 B/S 结构

B/S 结构是软件系统体系结构，通过它可以充分利用两端硬件环境的优势，将任务合理分配到 Client 端和 Server 端来实现，降低了系统的通讯开销。目前大多数应用软件系统都是 Client/Server 形式的两层结构，由于现在的软件应用系统正在向分布式的 Web 应用发展，Web 和 Client/Server 应用都可以进行同样的业务处理，应用不同的模块共享逻辑组件；因此，内部的和外部的用户都可以访问新的和现有的应用系统，通过现有应用系统中的逻辑可以扩展出新的应用系统。

根据实际开发的需要，位置信息管理系统设计的开发中选择了 B/S 结构开发 web 应用程序，通过浏览器访问服务器的的方式实现远程操作和数据共享。

### 2.3.2 Ajax 技术

AJAX 即“Asynchronous JavaScript and XML”（异步的 JavaScript 与 XML 技术），指的是一套综合了多项技术的浏览器端网页开发技术。Ajax 的概念由杰西·詹姆士·贾瑞特所提出。

传统的 Web 应用允许用户端填写表单（form），当提交表单时就向网页服务器发送一个请求。服务器接收并处理传来的表单，然后送回一个新的网页，但这个做法浪费了许多带宽，因为在前后两个页面中的大部分 HTML 码往往是相同的。由于每次应用的沟通都需要向服务器发送请求，应用的回应时间依赖于服务器的回应时间。这导致了用户界面的回应比本机应用慢得多。

与此不同，AJAX 应用可以仅向服务器发送并取回必须的数据，并在客户端采用 JavaScript 处理来自服务器的回应。因为在服务器和浏览器之间交换的数据大量减少（大约只有原来的 5%）[来源请求],服务器回应更快了。同时，很多的处理工作可以在发出请求的客户端机器上完成，因此 Web 服务器的负荷也减少了。

类似于 DHTML 或 LAMP，AJAX 不是指一种单一的技术，而是有机地利用了一系列相关的技术。虽然其名称包含 XML，但实际上数据格式可以由 JSON 代替，进一步减少数据量，形成所谓的 AJAJ。而客户端与服务器也并不需要异步。一些基于 AJAX 的“派生 / 合成”式（derivative/composite）的技术也正在出现，如 AFLAX。

# 第 3 章 系统需求分析

## 3.1 系统业务描述

总体上，系统的目标是帮助汽车部件生产企业对汽车部件，部件件的采购，销售进行统一管理，提高工作效率，便于公司的管理，为业务过程提供更加快捷、更好和更加经济的服务。

用户通过输入用户名和密码进入管理页面，然后进行业务流程处理。首先进行管理员登录，然后在后台对管理员类型进行判断，如果是仓库管理员，则会进入仓库管理员首页，可以对汽车部件信息，供应商信息，需求商信息，销售员和采购员信息经行管理，以及对销售员和采购员的操作信息的查看。如果是销售员/采购员，则会进入出库、入库首页，完成对汽车部件的出库入库操作，以及对供应商的添加。

## 3.2 可行性分析

可行性分析应从经济可行性，技术可行性和操作可行性三个方面考虑，以下是对本系统具体的可行性分析。

### 3.2.1 经济可行性分析

本系统的开发对硬件的要求也不高，无论从软件到硬件，开发成本都极低。同时，采用原型法的开发方法来开发本系统，开发时间短，能快速形成一个系统原型，最后再根据用户的需求加以改进，整个过程对人力、财力要求也不高，而且为企业节省了一个数据统计人员，每年让公司少五万元的支出，所以在经济上是可行的。

### 3.2.2 技术可行性分析

系统采用 B/S 架构，前台采用 html+css+js 编码实现页面的可视化与交互性，后端使用 C#实现对 SQL Server 数据库进行数据的存储、修改、删除等操作,该数据库具有方便、灵活的特点，适应该系统的开发。本系统要求的硬件标准不高，一般的硬件设备足够运行系统。

### 3.2.3 操作可行性分析

本系统界面友好，操作简单，任何人都可以很快掌握操作原理并使用。

综上所述，开发汽车部件仓储管理系统设计在经济、技术上、操作上都是可行的。

## 3.3 功能需求分析

汽车部件仓储信息管理系统的使用和相关者有：

1. 仓库管理员：仓库管理员作为公司的高级管理人员，可以对部件信息，供应商信息，需求商信息，采购员和销售员的信息的增、删、改、查以及对操作日志的查看。

2. 销售员/采购员：销售员/采购员作为公司的操作人员，主要进行入库、出库操作。同时，可以对需求商的信息进行查询，得到是否有需求商正等待发货而进行出库，也可以添加新的需求商用户。

仓库管理员作为一级管理员，主要的功能如下：

（1）汽车部件信息的管理

（2）供应商信息的管理

（3）需求商信息的管理

（4）采购员和销售员信息的管理

（5）采购员和销售员的操作日志查看

采购员/销售员作为二级管理员，主要的功能如下：

（1）汽车部件的出库、入库操作

（2）需求商的信息的添加

## 3.4 分析模型

### 3.4.1 仓储管理的工作流程图

工作流（Workflow），指“业务过程的部分或整体在计算机应用环境下的自动化”。工作流主要解决的主要问题是：为了实现某个业务目标，利用计算机在多个参与者之间按某种预定规则自动传递文档、信息或者任务。

1. 工作流程管理

数据管理工作流程图如图 3-1 所示：

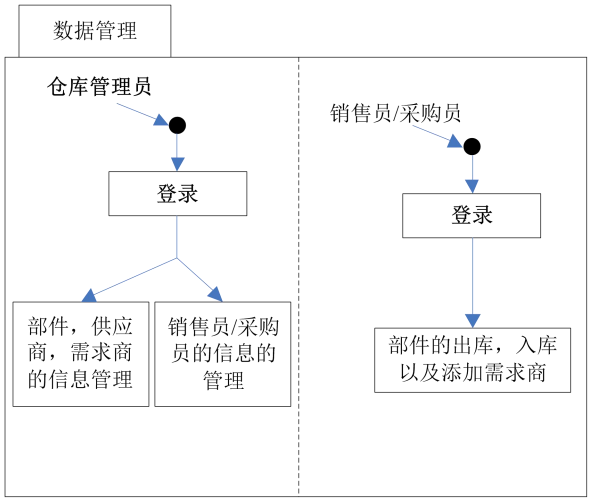



图 3-1 数据管理工作流程图

2. 汽车部件信息查看

汽车部件信息查看工作流程图如图 3-2 所示：

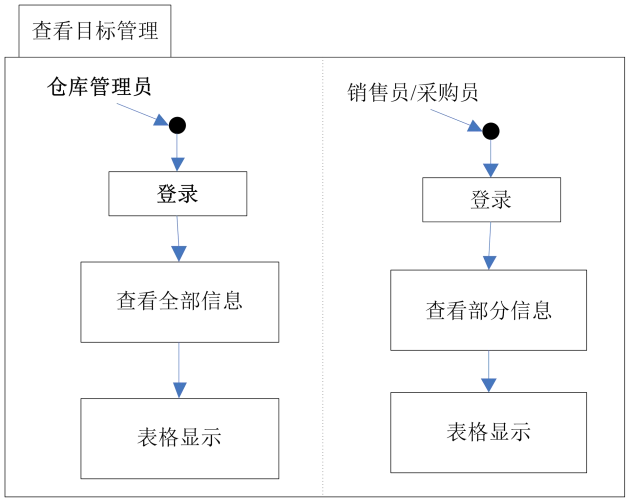
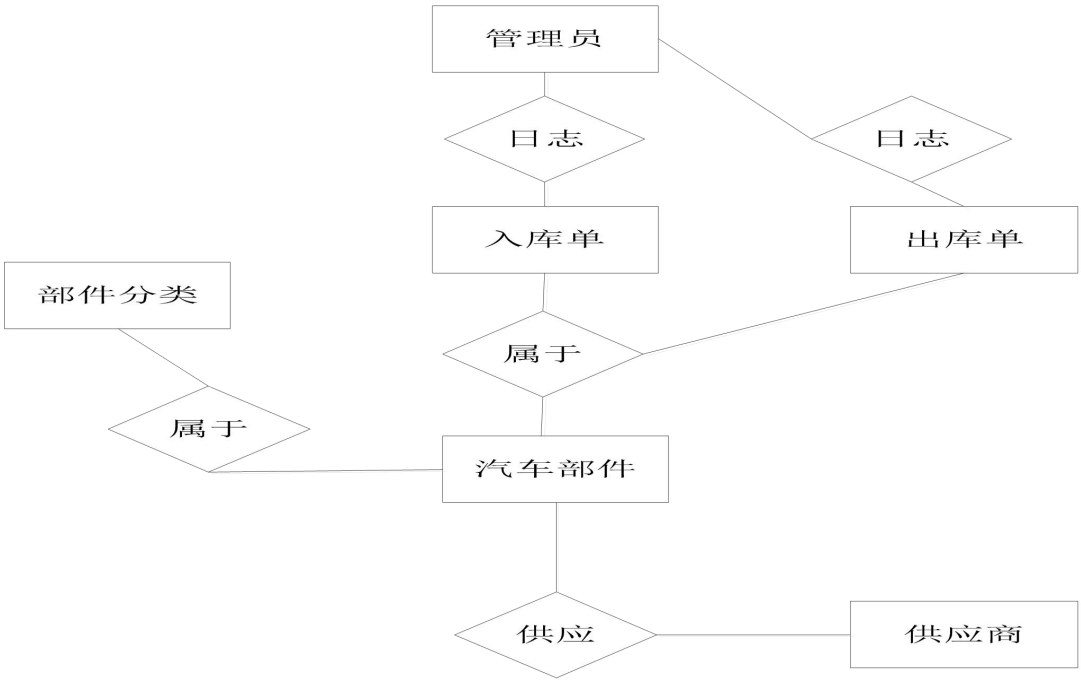



图 3-2 部件信息查看工作流程图

### 3.4.2 实体模型图（ER 图）

E-R 图也称实体-联系图(Entity Relationship Diagram)，提供了表示实体类型、属性和联系的方法，用来描述现实世界的概念模型。
系统总 ER 图如图 3-3 所示：




3-3 系统总 ER 图

管理员实体用来存储管理员的基本信息，其实体图如图 3-4 所示：

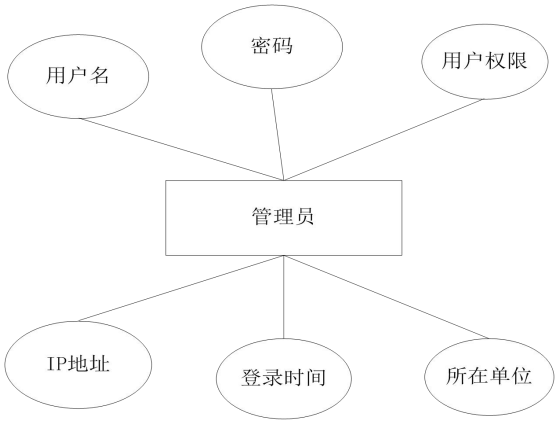



图 3-4 管理员实体图

供应商实体用来存储供应单位的相关信息，其实体图如图 3-5 所示：

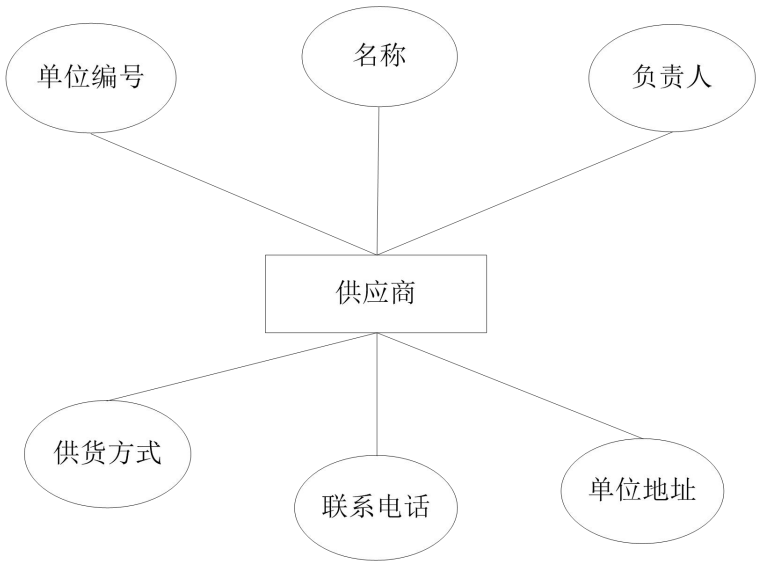



图 3-5 供应商实体图

需求商实体用来存储需求单位的相关信息，其实体图如图 3-6 所示：

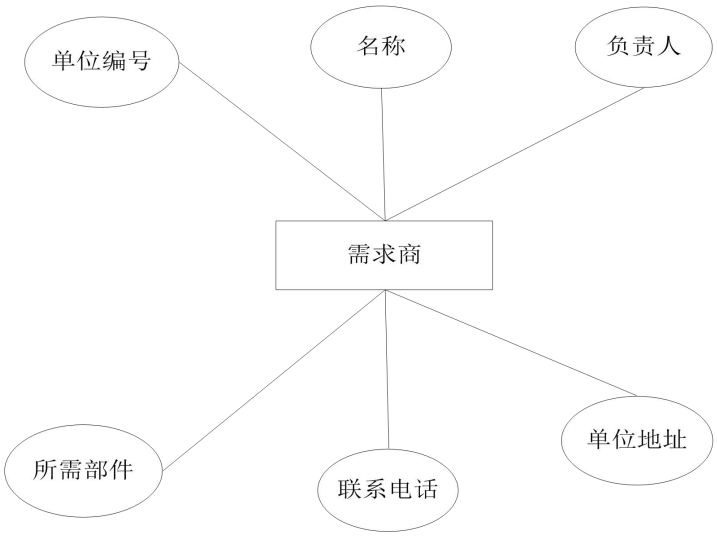



图 3-6 需求商实体图

汽车部件实体存储汽车部件的相关信息，其实体图如图 3-7 所示：

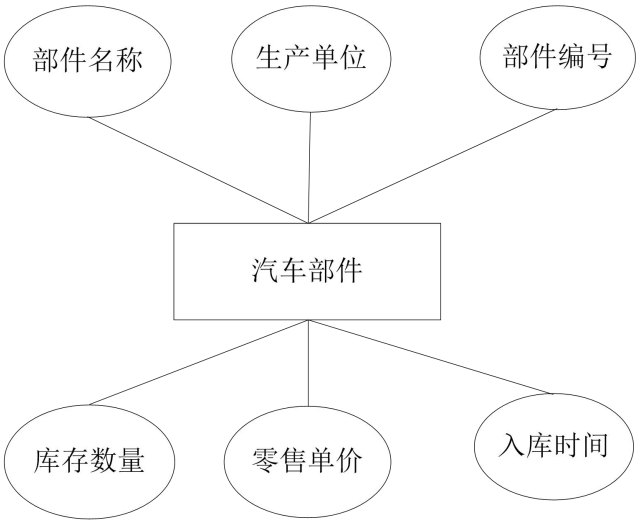



图 3-7 部件实体图

操作日志实体存储相关操作的信息，其实体图如图 3-8 所示：

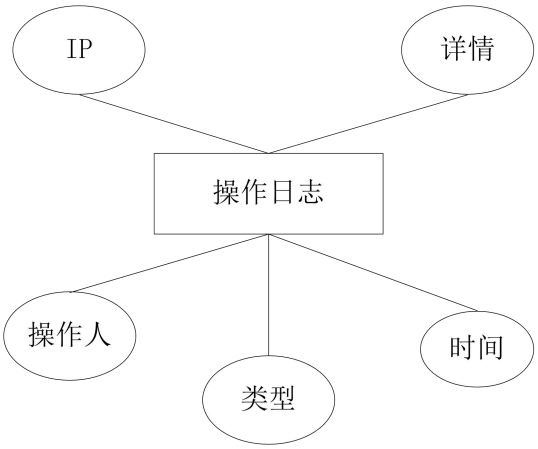



图 3-8 操作日志实体图

汽车部件的分类实体图如图 3-9 所示：

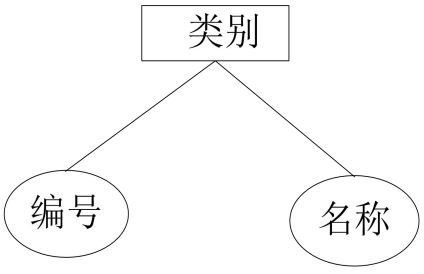



图 3-9 汽车部件分类实体图

出库单的信息，其实体图如图 3-10 所示：

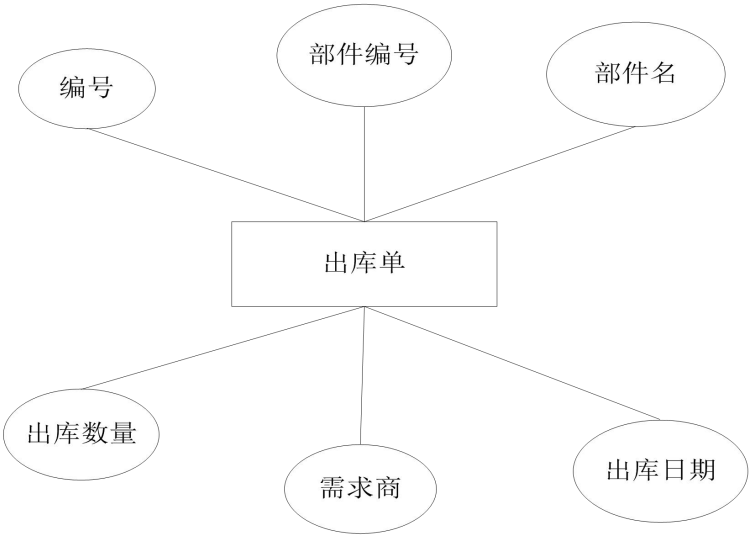



图 3-10 出库单实体图

入库单的信息，其实体图如图 3-11 所示：

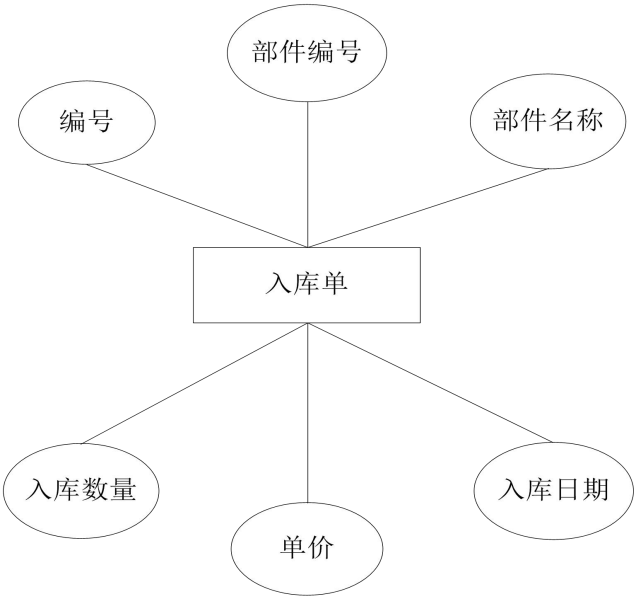



图 3-11 入库单实体实体图

### 3.4.3 系统用例图

用例图是指由参与者（Actor）、用例（Use Case），边界以及它们之间的关系构成的用于描述系统功能的视图。用例图（User Case）是外部用户（被称为参与者）所能观察到的系统功能的模型图。用例图呈现了一些参与者，一些用例，以及它们之间的关系，主要用于对系统、子系统或类的功能行为进行建模。

此系统的系统用例图如图 3-12 所示：

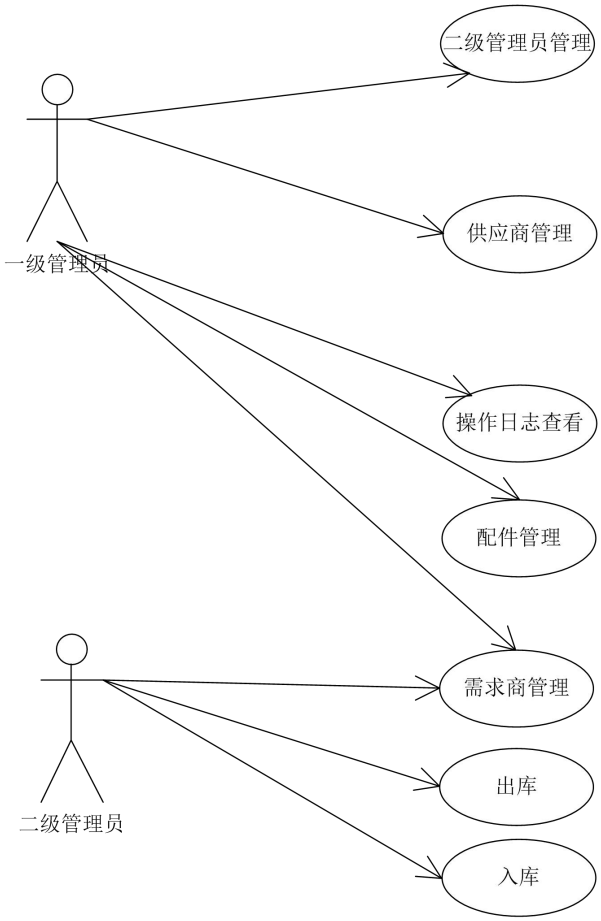



图 3-12 系统用例图

# 第 4 章 系统设计

## 4.1 概要设计

系统的各部分功能模块独立开发、调试,然后利用系统集成的方法将各个模块信息传入数据库。各个功能模块采用事件驱动的方式 与应用程序进行交互,系统部分程序的应用执行是在后台进行的。汽车部件仓储管理系统设计的系统总体结构设计如图 4-1 所示。


图 4-1 汽车部件信息管理系统设计功能模块图

一级管理员管理模块有如下功能：一级管理员可对该模块进行管理，可以查看所有汽车部件，供应商，需求商，管理员的信息，并且可以对信息进行添加，修改，删除，而且可以查看日志信息。其模块功能图如图 4-2 所示：

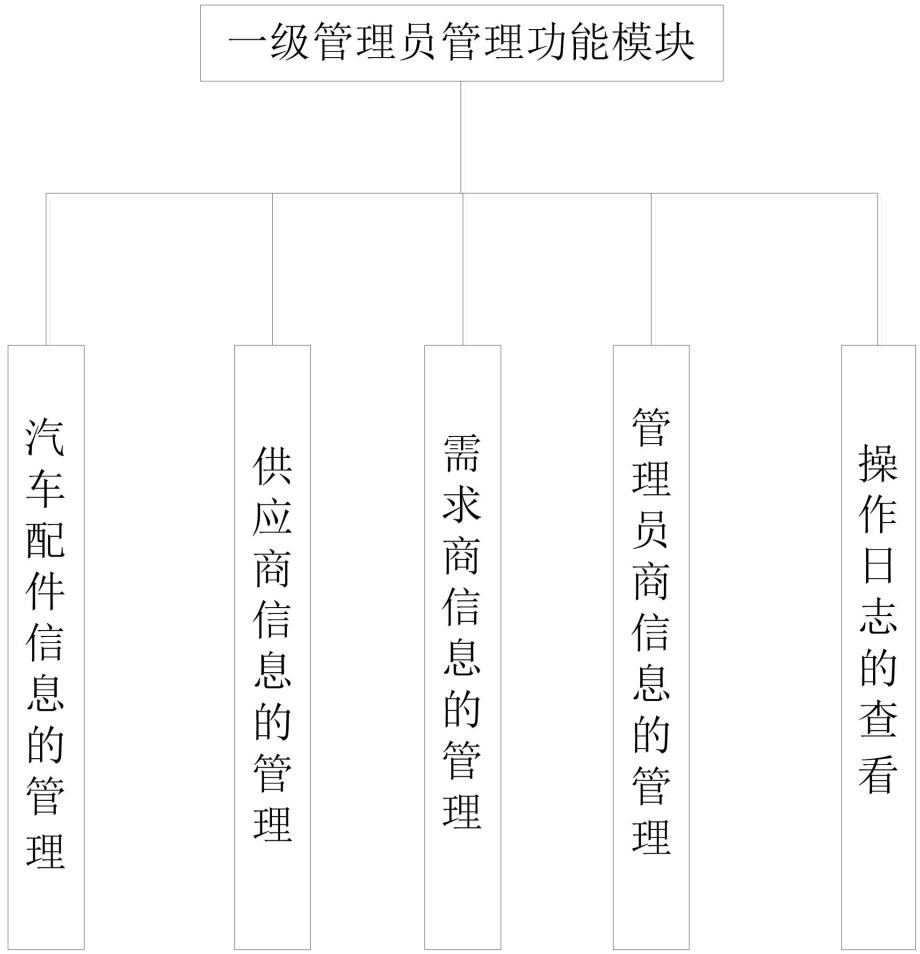



图 4-2 一级管理员管理功能模块图

## 4.2 数据库设计

通过以上的对系统的综合分析，本系统选择 SQL Server 作为系统的数据库，数据库中包括管理员信息、供应商基本信息、需求商基本信息、二级管理员信息、操作日志信息。

管理员表：主要用于存储验证登陆的管理员信息。管理人员密码登陆系统进行操作。管理员信息表如表 4-1 所示。

表 4-1 管理员信息表

| 字段名称 | 数据类型 | 允许为空 | 主键 | 备注 |
|---|---|---|---|---|
| UserId | int | 否 | 是 | 用户 ID |
| UserName | nvarchar(50) | 否 | 否 | 用户名 |
| Password | nvarchar(50) | 否 | 否 | 密码 |
| Type | nvarchar(50) | 是 | 否 | 用户类型 |
| LoginTime | Datetime | 是 | 否 | 登录时间 |
| Department | nvarchar(50) | 是 | 否 | 所属部门 |

汽车部件信息表：主要存储汽车部件的基本信息，用于二级管理员的出库入库以及一级管理员对部件信息的修改。汽车部件信息表如表 4-2 所示。

表 4-2　汽车部件信息表

| 字段名称 | 数据类型 | 允许为空 | 主键 | 备注 |
|---|---|---|---|---|
| PartId | int | 否 | 是 | 部件 ID |
| PartName | nvarchar(50) | 是 | 否 | 部件名称 |
| FromDepartId | nvarchar(50) | 是 | 否 | 供应商 ID |
| Num | nvarchar(50) | 是 | 否 | 编号 |
| InTime | Datetime | 是 | 否 | 入库时间 |
| UnitPrice | decimal(18, 2) | 是 | 否 | 单价 |
| Quantity | int | 是 | 否 | 数量 |
| TypeId | Int | 是 | 否 | 类别 ID |

供应单位信息表：主要存储供应单位的基本信息，用于一级管理员对供应商的基本信息的查看以及修改。供应单位信息表如表 4-3 所示。

表 4-3 供应单位信息表

| 字段名称 | 数据类型 | 允许为空 | 主键 | 备注 |
|---|---|---|---|---|
| Id | int | 否 | 是 | 供应商 ID |
| Name | nvarchar(50) | 否 | 否 | 供应商名称 |
| Principal | nvarchar(50) | 是 | 否 | 负责人 |
| Address | nvarchar(50) | 是 | 否 | 地址 |
| Phone | nvarchar(50) | 是 | 否 | 电话 |
| Ways | nvarchar(50) | 是 | 否 | 运输方式 |
| Num | nvarchar(50) | 否 | 否 | 编号 |

需求单位信息表：主要存储需求单位的基本信息，用于一级管理员对需求商的基本信息的查看以及修改。需求单位信息表如表 4-4 所示。

表 4-4 需求单位信息表

| 字段名称 | 数据类型 | 允许为空 | 主键 | 备注 |
|---|---|---|---|---|
| Id | int | 否 | 是 | ID |
| Num | nvarchar(50) | 否 | 否 | 编号 |
| Name | nvarchar(50) | 否 | 否 | 名称 |
| Principal | nvarchar(50) | 是 | 否 | 负责人 |
| Phone | nvarchar(50) | 是 | 否 | 电话 |
| IsDeliver | bit | 是 | 否 | 是否已发货 |
| Address | nvarchar(50) | 是 | 否 | 地址 |
| NeedPartName | nvarchar(50) | 是 | 否 | 所需配件 |
| NeedNum | int | 是 | 否 | 所需编号 |

操作日志信息表：主要存储二级管理员的操作的基本信息，一级管理员对二级管理员操作信息进行查看。需求单位信息表如表 4-5 所示。

表 4-5 操作日志信息表

| 字段名称 | 数据类型 | 允许为空 | 主键 | 备注 |
|---|---|---|---|---|
| Id | int | 否 | 是 | ID |
| Operator | nvarchar(50) | 否 | 否 | 操作人 |
| Time | Datetime | 否 | 否 | 操作时间 |
| Details | Nverchar(1000) | 是 | 否 | 细节 |
| Type | nvarchar(50) | 是 | 否 | 操作类型 |
| IP | nvarchar(50) | 是 | 否 | 操作人 IP |

汽车部件分类信息表：主要功能是将汽车部件进行分类。汽车部件分类信息表如表 4-6 所示。

表 4-6 部件类别信息表

| 字段名称 | 数据类型 | 允许为空 | 主键 | 备注 |
|---|---|---|---|---|
| TypeId | int | 否 | 是 | ID |
| TypeName | nvarchar(50) | 是 | 否 | 类型名称 |

入库单信息表：主要记录二级管理员的入库记录。入库单信息表如表 4-7 所示。

表 4-7 入库单信息表

| 字段名称 | 数据类型 | 允许为空 | 主键 | 备注 |
|---|---|---|---|---|
| InNum | nvarchar(50) | 否 | 是 | 入库单编号 |
| PNum | nvarchar(50) | 是 | 否 | 部件编号 |
| PName | nvarchar(50) | 否 | 否 | 部件名称 |
| UnitPrice | decimal(18, 2) | 是 | 否 | 单价 |
| InQuantity | Int | 是 | 否 | 入库数量 |
| InTime | datetime | 是 | 否 | 入库时间 |

出库单信息表：主要记录二级管理员的出库记录。出库单信息表如表 4-8 所示。

表 4-8 入库单信息表

| 字段名称 | 数据类型 | 允许为空 | 主键 | 备注 |
|---|---|---|---|---|
| OutNum | nvarchar(50) | 否 | 是 | 出库单编号 |
| PNum | nvarchar(50) | 是 | 否 | 部件编号 |
| PName | nvarchar(50) | 否 | 否 | 部件名称 |
| NeedMerchant | nvarchar(50) | 是 | 否 | 需求商 |
| OutQuantity | Int | 是 | 否 | 出库数量 |
| OutTime | datetime | 是 | 否 | 出库时间 |

## 4.3 功能模块设计流程图

位置信息管理系统设计其功能已详细介绍，本节将详细介绍各个功能模块的设计流程图。其中登录界面程序流程图，如图 4-3 所示：

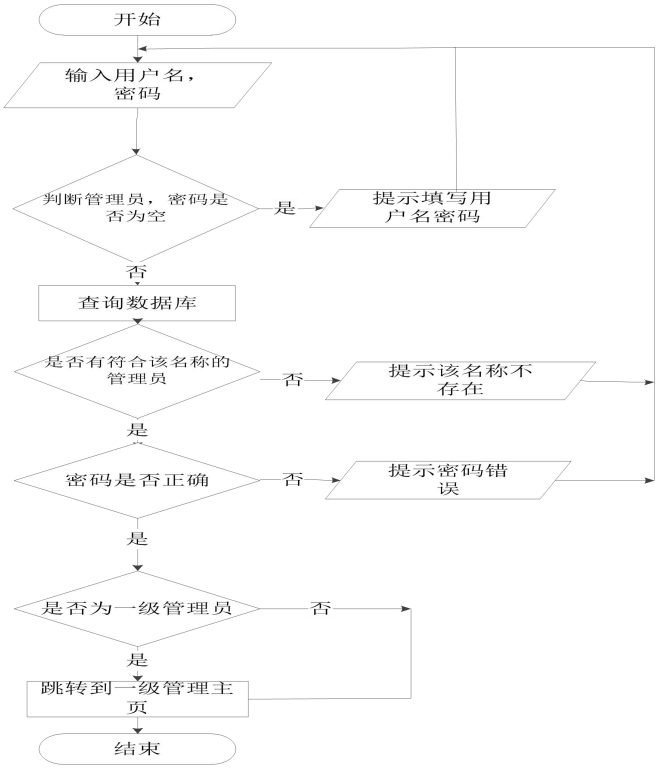



图 4-3 登录页面程序流程图

系统中很多功能都涉及到修改数据，修改数据信息主要是实现前端数据与后台数据库的交互，以 ID 为判断条件将修改后的信息用 SQL 语句调用数据库修改相应的数据，刷新数据库及相应页面，显示数据库的信息。修改数据流程图如图 4-4 所示，

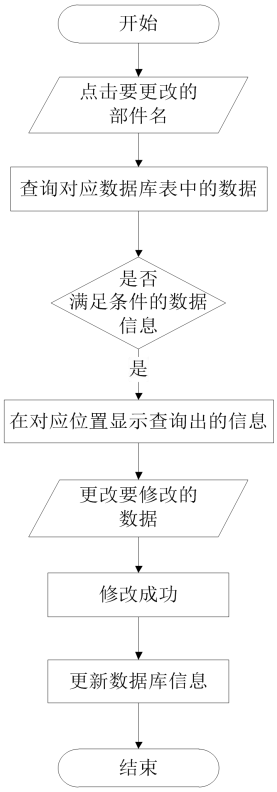



图 4-4 修改汽车部件信息程序流程图

删除数据信息，实现主要是根据删除按钮或是全选删除按钮的点击事件获取相应的要删除信息的 ID，然后根据 ID 利用 SQL 语句删除数据库中的数据，更新数据库。删除程序主要流程图，如图 4-5 所示：

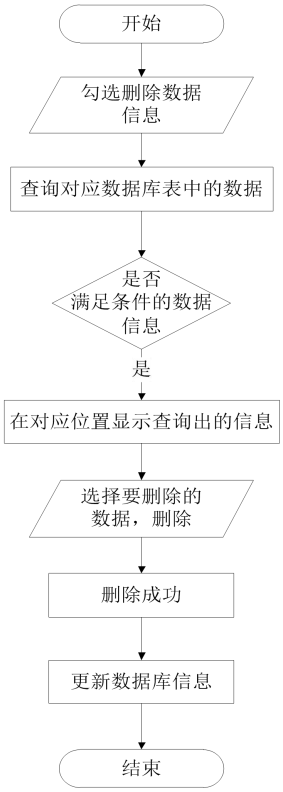



图 4-5 删除汽车部件信息程序流程图

增加数据信息的功能主要是将表单中的数据提交到后台数据库中，在由后台对数据做相应的判断，若不符合要求，则显示增加不成功，不保存到数据库，否则将数据保存到相应数据库。增加数据信息的程序流程图，如图 4-6 所示：

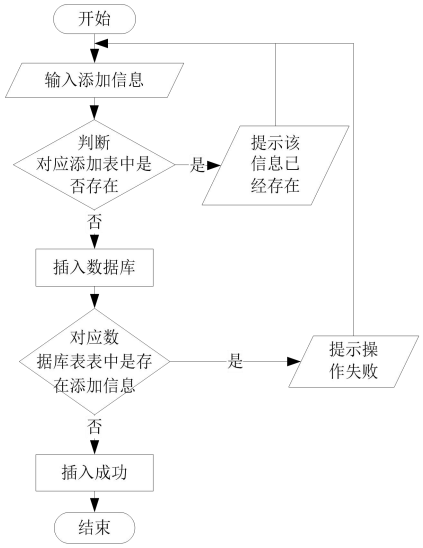



图 4-6 增加汽车部件信息信息程序流程图

# 第 5 章 系统实现与测试

## 5.1 系统实现

### 5.1.1 数据库连接

<add
name="MainConn"connectionString="server=.;database=CarParts;user=sa;pwd=0301;Max Pool Size=512;" provider Name="System.Data.SqlClient"/>

### 5.1.2 登录功能

通过登录界面，判断管理员账户密码，而且区分出仓库总管和采购员/销售员，分别进入仓库总管首页和采购员/销售员首页，登录界面如图 5-1 所示：


图 5-1 登录界面

实现登录功能的前端核心代码如下所示：

```
<form>
                <label for="username">用户名</label>
                <input name="username" type="text" placeholder="请输入用户名" id="name"/>
                <label for="pass">密码</label>
                <input name="pass" type="password" placeholder="请输入密码" id="password"/>
                <input value="登录" id="submit" />
</form>
```

实现登录功能的后台核心代码如下所示：

```
            //判断表中是否存在数据，如果有数据则登录成功，如果没有数据，则登录失败
            if (ds.Tables[0].Rows.Count > 0)
            {
                //根据判断选择进入哪个主页,无论进来的管理员或者是普通用户都记录下操作者的 ID，方便记录日志
                if (ds.Tables[0].Rows[0]["Type"].ToString() == "1")
                {
                    context.Session["Id"] = ds.Tables[0].Rows[0]["UserId"].ToString();
                    context.Session["LoginTime"] = DateTime.Now.ToString();
                    context.Response.Write("ok1:登录成功");
                }
```

```
                else
                {
                    context.Session["Id"] = ds.Tables[0].Rows[0]["UserId"].ToString();
                    context.Session["LoginTime"] = DateTime.Now.ToString();
                    context.Response.Write("ok2:登录成功");
                }
            }
            //验证失败弹出提示框
            else
            {
                context.Response.Write("用户名或密码错误，请重新登录");
            }
        }
```

### 5.1.3 仓库管理员主要功能设计

仓库管理员主要是对部件信息，供应商信息，需求商信息，采购员和销售员信息的管理，以及对采购员和销售员的操作日志的查看，如图 5-2 所示：


图 5-2 一级管理员首页

界面的核心代码如下所示:

```
<%--核心内容展示区域--%>
        <div data-options="region:'center',title:'信息管理'" class="right">
            <div class="easyui-tabs" style="width:700px;height:250px;" class="right" fit="true" id="tt">
                <div title="部件信息" style="padding:10px;overflow:hidden;"class="right">
                    <iframe src="InfoPage/PartsInfo.aspx" scrolling="no" width="100%" height="100%" frameborder="0"></iframe>
                </div>
            </div>
        </div>
        <%--核心内容展示区域结束--%>
```

在仓库管理员的界面里，如果点击部件名称，下面的这么界面就会显示该部件的相关信息，并且可以对相关信息进行修改，如果点击添加按钮，就会显示添加部件信息界面，并且可以完成部件的添加，如图 5-3 所示：


图 5-3 添加部件信息界面

### 5.1.4 采购员/销售员主要功能设计

通过登录界面，会经过判断进入不同的管理员界面，其中采购员/销售员主要进行出库入库操作，根据下拉选择出库/入库，然后通过输入数量来控制批量操作，其中出入/入库首页如图 5-4 所示：


图 5-4 二级管理员主界面

出入/入库首页界面核心代码如下所示：

```
<asp:GridView ID="gvNeedMer" runat="server" AutoGenerateColumns="False" DataKeyNames="Id"
AllowPaging="True" OnPageIndexChanging="gvParts_PageIndexChanging">
    <Columns>
        <asp:CheckBoxField DataField="IsDeliver" HeaderText="已发货" />
```

```
                <asp:BoundField DataField="Num" HeaderText="编号" />
                <asp:TemplateField HeaderText="单位名称">
                    <ItemTemplate>
                            <a id="linkToPartInfoEdit" target="_blank"
href="javascript:linkToPart(<%# Eval("Id") %>)">
                            <%#Eval("Name") %>
                        </a>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:BoundField DataField="Principal" HeaderText="负责人" />
                <asp:BoundField DataField="Address" HeaderText="地址" />
                <asp:BoundField DataField="Phone" HeaderText="联系电话" />
                <asp:TemplateField HeaderText="添加">
                    <ItemTemplate>
                        <input type="button" id="btnAdd" value="添加"/>
                    </ItemTemplate>
                </asp:TemplateField>
            </Columns>
        </asp:GridView>
    </div>
    </form>
```

入库界面完成汽车部件的入库功能，其中的条数是根据主页输入的所需入库条数而设定，主要实现了汽车部件的数量的更改以及汽车部件的单价的更新，如图 5-5 所示：


图 5-5 入库界面

系统设计内容除了以上的功能，还有两个核心功能，第一个是进行数据绑定的方法，核心代码如下所示：

```
    public void BindData()
    {
        //查询数据
        StringBuilder sb = new StringBuilder();
        sb.AppendLine("SELECT
```

```
C.PartId,C.Num,C.PartName,S.Name,C.InTime,C.Quantity,C.UnitPrice");
            sb.AppendLine("FROM CarParts AS C INNER JOIN Supplies AS S");
            sb.AppendLine("ON C.FromDepartId = S.Id;");
            DataSet ds = DbHelperSQL.Query(sb.ToString());
            //将数据绑定到 gvParts 上展示
            gvParts.DataSource = ds.Tables[0];
            gvParts.DataBind();
```

第二个是获取 ip 的方法，核心代码如下所示：

```
    //获取 Ip
    public string GetIP()
    {
        string result = String.Empty;
        result = HttpContext.Current.Request.ServerVariables["HTTP_X_FORWARDED_FOR"];
        if (string.IsNullOrEmpty(result))
        {
            result = HttpContext.Current.Request.ServerVariables["REMOTE_ADDR"];
        }
        if (string.IsNullOrEmpty(result))
        {
            result = HttpContext.Current.Request.UserHostAddress;
        }
        if (string.IsNullOrEmpty(result))
        {
            return "127.0.0.1";
        }
        return result;
    }
```

## 5.2 系统测试

为了保证设计完成后的软件是有效的，健壮的，在软件开发的生命周期中要对软件进行测试，测试工作可以验证软件的需求是否都得以实现，软件是否正 确地提供了需要的功能，以及软件是否能健壮稳定地运行。所以，本系统的测试主要以如下几方面为切入点：

一、功能验收测试：对应软件的需求分析和详细设计文档，检查系统所应该实现的功能是否已经实现。对于一个软件，完成并能够使用的最基本条件是所有的功能都能覆盖，并且功能能够成功执行。

二、集成验收测试：在保证了每个功能都实现并可用之后，需要验证系统的 每个功能的正确性，于是进一步的测试则要保证每一个功能的每种可能执行方式 都能得到正确的结果。这样我们才可以说，对外发布的系统是一个正确的系统。

三、健壮性、稳定性及性能测试：为了让系统能够稳定的高效运行，需要对 系统进行以下的测试。如，选择一些非正常的输入，这时系统需要能继续稳定运 行，或者很容易从错误中自动恢复；当长时间的进行持续的操作时，系统对资源 的消耗应该处于稳定的，可接受的范围内，尤其要避免内存泄露等问题带来的风 险；系统还要达到需求分析文档中所规定的性能指标，所以还要针对实际情况进 行性能测试；最后系统还

要根据需要对不同平台、不同环境的兼容性进行测试。 整个系统需要测试的模块主要有登录模块，一级管理员的管理模块，二级管理员的管理模块。

系统测试：功能测试、性能测试、验收测试。目的是为了保证所实现的系统确实是用户所需要的。

1.登录验证，当用户名或密码不输入时提示用户重新输入，以及当用户输入的用户名和密码不匹配时，提示重新输入。当输入的是一级管理员则进入一级管理员首页，当输入的是二级管理员进入二级管理员首页。如图 5-6 所示：


图 5-6 判断是否输入用户名界面


图 5-7 判断是否输入密码界面


图 5-8 判断用户名与密码是否正确界面

2.修改部件信息验证，当管理员点击部件名称的时候，将进行对部件信息的修改，弹出界面如图 5-9 所示：

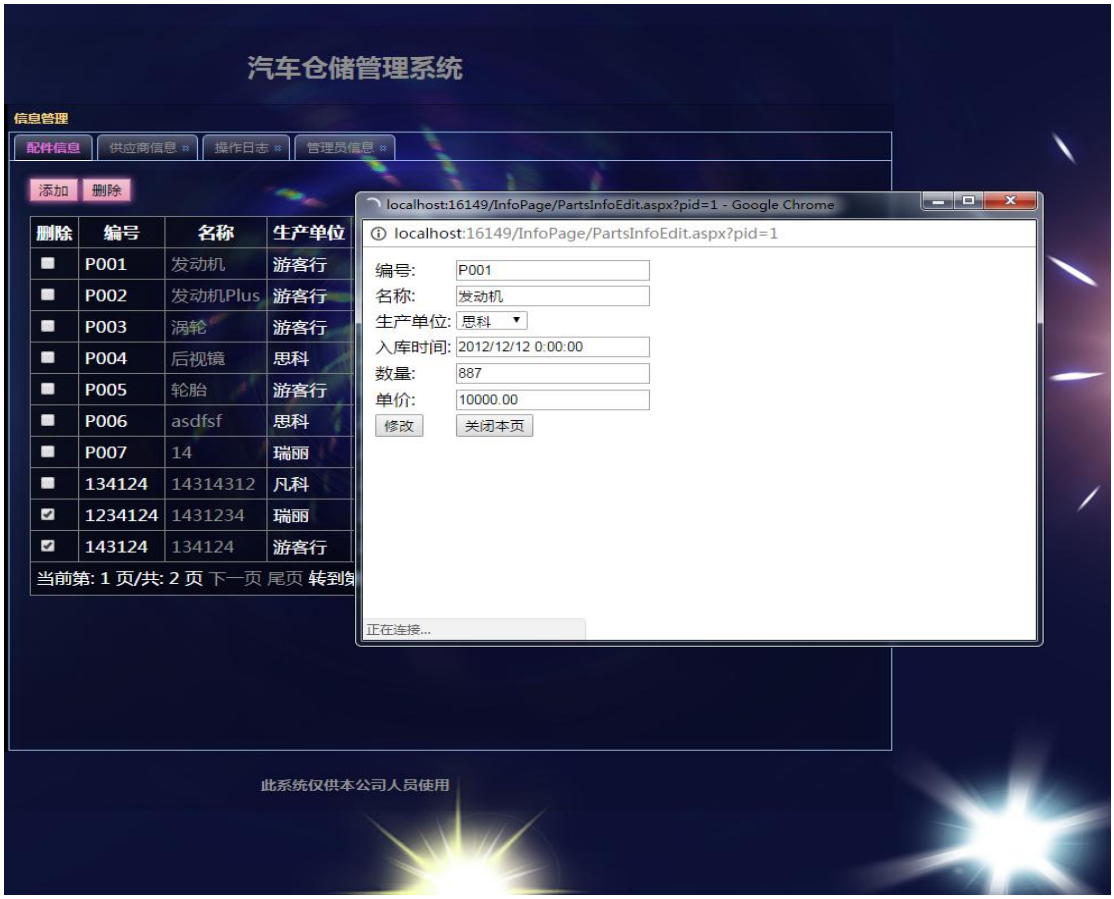



图 5-9 部件信息的修改界面

3.删除部件信息验证，当管理员勾选删除的勾选框，然后点击删除按钮，便对信息进行删除，并刷新界面，如图 5-10 所示：


图 5-10 部件信息的删除界面

2.入库验证，当二级管理员选择部件编号，然后输入入库的数量，点击确认，并刷新界面，如图 5-11 所示：


图 5-11 入库界面

# 结论与展望

汽车部件管理系统设计终于完成了，在完成设计及开发的过程中，我遇到了很多的问题也学习了很多的东西。同时在解决处理问题的过程中渐渐实践了一些作为程序员应该注意的事项。其中有些较为重要的细节如下：

在编码过程中注意添加注释和排版，方便自己以及其他查看程序的人员更加快速便捷得查找问题错误以及关键字段功能，提高效率。

可以从查阅资料中学习借鉴别人的编程思路，以及在编程中遇到的问题的解决办法和原因，再通过实战编程努力丰富自己的经验。

在设计界面的过程中，可根据需要设计不同的界面效果。满足客户的需求和大众的审美才是最重要的。

程序功能的调试是检验程序主要功能是否满足要求的过程，即使简化辅助功能也要使主要功能尽可能完善。

本次毕业设计给了我一个实践的机会，锻炼了自己的意志品质和适应社会的能力，而且在真正的实践中加深了对理论知识的理解，积累了编程经验，为毕业走向社会打下了良好的基础。

但是由于时间的仓促和工作经验的不足及其他方面的原因，在软件设计中实现的功能有待完善的地方：

1、在功能上，还有需要进行扩展的地方。

2、在界面上，还不够大气，有点个人色彩。

3、在用户体验上，缺少更多的交互。

汽车部件管理从某种角度来说，发展潜力还是很大的。一方面，近几年随着汽车在数量上的不断增加，促使市场上对汽车部件的需求量也在逐年加大。另一方面，计算机科技的发展，在这种科技领域必然会代替很多人工的劳动。

# 致谢

时光匆匆如流水，岁月如歌，白驹过隙，转眼便是大学毕业时节，春梦秋云，聚散匆匆,离校日期已日趋渐进。回首这四年的求学历程，对那些引导我、帮助我、激励我的人，我心中充满了感激。经历了找工作的喧嚣与坎坷，我明白了对于要进入社会的我们来说工作又是一个学习进步，优胜劣汰的过程。

感谢四年里相处过的同学们，感谢你们曾经的鼓励、关心和帮助，在生活和学习中，有了你们，使得众多难题，迎刃而解，同窗之谊，我将终生难忘！

感谢我的家人，在我漫长的求学旅途中，你们是我的坚强后盾，不断前进的动力，时刻给我的支持与鼓励。

最后，诚挚的感谢我的指导老师王勇老师。他在忙碌的教学工作中挤出时间来审查、修改我的论文。还有教过我的所有老师们，你们严谨细致、一丝不苟的作风一直是我工作、学习中的榜样;他们循循循善诱的教导和不拘一格的思路给予我无尽的启迪。

爱读书，爱运动，爱生活,这是我在求职简历上的自我评价。就用这话作为这篇论文的一个结尾。这是我对人生的态度，积极向上，为自己的理想而努力奋斗，不屈不挠，前方的路还未明朗，但已经做好了接受一切的准备。不断学习，不断前行，不断提升自己。

作者：

2017 年　月　日

# 参考文献

[1] Karli Watson 著.C#入门经典（第 6 版）[M]，2014-8
[2] （美）内格尔（Nagel.C）等所著.C#高级编程[M]，2008-10-1
[3] 王珊,萨师煊.数据库系统概论（第四版）[M].北京：高等教育出版社，2006.5
[4] ASP.NET 入门经典(第 9 版)[M]，2016-11
[5] Baron Scbwartz.高性能 MySQL（第 3 版）[M].电子工业出版社，2013
[6] （美）加洛韦等著.ASP.NET MVC 5 高级编程(第 5 版)[M],2015-02
[7] 构建之法 现代软件工程（第二版）[M],2015-07
[8] 王国辉,王毅.数据库系统开发案例精选[M].人民邮电出版社,2006
[9] 软件工程：实践者的研究方法（原书第 8 版）,2016-11
[10] 程成，陈霞.软件工程[M].机械工业出版社，2003
[11] 于大伟,田地;基于 WEB 信息系统的优化管理及架构调整[J];长春工程学院学报(自然科学版);2006 年 01 期:54-56
[12] 汤星群.基于数字档案馆建设的两点思考[J].档案时空，2005
[13] 赵国玲著.网页制作与数据库应用[M]，2006.3
[14] 高文会著.Web 应用开发技术[M]，2005.8
[15] 黄贤英著.UML 建模过程及在需求分析中的应用[M]，2001.2
[15] Gary P Johnston，David V. Bowen.he benefits of electronic recordsmanagement systems: a general review of published and some unpublishedcases. RecordsManagement Journal，2005
[16] Keith Gregory.Implementing an electronic records management system: Apublic sector case study. Records Management Journal，2005
[17] Duranti Luciana.Concepts，Principles，and Methods for the Management of Electronic RecordsR[J].Information Society，2001

# 附　录

## 附录 A 外文文献及其译文

## Software and software engineering Software Requirements

## ——the software appearance and enumerates

As the decade of the 1980s began, a front page story in business week magazine trumpeted the following headline:” software: the new driving force.”software had come of age—it had become a topic for management concern. during the mid-1980s,a cover story in foreune lamented “A Growing Gap in Software,” and at the close of the decade, business week warned managers about”the Software Trap—Automate or else.”As the 1990s dawned , a feature story in Newsweek asked ”Can We Trust Our Software? ”and The wall street journal related a major software company’s travails with a front page article entitled “Creating New Software Was an Agonizing Task …” these headlines, and many others like them, were a harbinger of a new understanding of the importance of computer software ---- the opportunities that it offers and the dangers that it poses.

Software has now surpassed hardware as the key to the success of many computer-based systems. Whether a computer is used to run a business, control a product, or enable a system , software is the factor that differentiates . The completeness and timeliness of information provided by software (and related databases) differentiate one company from its competitors. The design and “human friendliness” of a software product differentiate it from competing products with an otherwise similar function .The intelligence and function provided by embedded software often differentiate two similar industrial or consumer products. It is software that can make the difference.

During the first three decades of the computing era, the primary challenge was to develop computer hardware that reduced the cost of processing and storing data .Throughout the decade of the 1980s,advances in microelectronics resulted in more computing power at increasingly lower cost. Today, the problem is different .The primary challenge during the 1990s is to improve the quality ( and reduce the cost ) of computer-based solutions- solutions that are implemented with software. In this subsection:The power of a 1980s-era mainframe computer is available now on a desk top. The awesome processing and storage capabilities of modern hardware represent computing potential. Software is the mechanism that enables us to harness and tap this potential.

The context in which software has been developed is closely coupled to almost five decades of computer system evolution. Better hardware performance, smaller size and lower cost have precipitated more sophisticated computer-based systems. We’re moved form vacuum tube processors to microelectronic devices that are capable of processing 200 million connections per second .In popular books on “the computer revolution,” Osborne characterized a “new industrial revolution,” Toffer called the advent of microelectronics part of “the third wave of change” in human history , and Naisbitt predicted that the transformation from an industrial society to an “information society” will have a profound impact on our lives. Feigenbaum and McCorduck suggested that information and knowledge

will be the focal point for power in the twenty-first century, and Stoll argued that the “electronic community” created by networks and software is the key to knowledge interchange throughout the world . As the 1990s began , Toffler described a “power shift” in which old power structures( governmental, educational, industrial, economic, and military) will disintegrate as computers and software lead to a “democratization of knowledge.”

Depicts the evolution of software within the context of. computer-based system application areas. During the early years of computer system development, hardware underwent continual change while software was viewed by many as an afterthought. Computer programming was a "seat-of-the-pants" art for which few systematic methods existed. Software development was virtually unmanaged--until schedules slipped or costs began to escalate. During this period, a batch orientation was used for most systems. Notable exceptions were interactive systems such as the early American Airlines reservation system and real-time defense-oriented systems such as SAGE. For the most part, however, hardware was dedicated to the union of, a single program that in turn was dedicated to a specific application.

During the early years, general-purpose hardware became commonplace. Software, on the other hand, was custom-designed for each application and had a relatively limited distribution. Product software(i.e., programs developed to be sold to one or more customers) was in its infancy . Most software was developed and ultimately used by the same person or organization. You wrote it, you got it running , and if it failed, you fixed it. Because job mobility was low , managers could rest assured that you’d be there when bugs were encountered.

Because of this personalized software environment, design was an implicit process performed in one’s head, and action was often nonexistent. During the early years we learned much about the implementation of computer-based systems, but relatively little about computer system engineering .In fairness , however , we must acknowledge the many outstanding computer-based systems that were developed during this era. Some of these remain in use today and provide landmark achievements that continue to justify admiration.

The second era of computer system evolution (Figure 1.1) spanned the decade from the mid-1960s to the late 1970s. Multiprogramming and multiuse systems introduced new concepts of human-machine interaction. Interactive techniques opened a new world of applications and new levels of hardware and software sophistication . Real-time systems could collect, analyze, and transform data form multiple sources , thereby controlling processes and producing output in milliseconds rather than minutes . Advances in on-line storage led to the first generation of database management systems.

The second era was also characterized by the use of product software and the advent of "software houses." Software was developed for widespread distribution in a multidisciplinary market. Programs for mainframes and minicomputers were distributed to hundreds and sometimes thousands of users. Entrepreneurs from industry, government, and academia broke away to "develop the ultimate software package" and earn a bundle of money.

As the number of computer-based systems grew, libraries of computer software began to expand. In-house development projects produced tens of thousands of program source statements. Software products purchased from the outside added hundreds of thousands of new statements. A dark cloud appeared on the horizon. All of these programs--all of these

source statements-had to be corrected when faults were detected, modified as user requirements changed, or adapted to new hardware that was purchased. These activities were collectively called software maintenance. Effort spent on software maintenance began to absorb resources at an alarming rate.

Worse yet, the personalized nature of many programs made them virtually unmentionable. A "software crisis" loomed on the horizon.

The third era of computer system evolution began in the mid-1970s and continues today. The distributed system--multiple computers, each performing functions concurrently and communicating with one another- greatly increased the complexity of computer-based systems. Global and local area networks, high-bandwidth digital communications, and increasing demands for 'instantaneous' data access put heavy demands on software developers.

The third era has also been characterized by the advent and widespread use of microprocessors, personal computers, and powerful desk-top workstations. The microprocessor has spawned a wide array of intelligent products-from automobiles to microwave ovens, from industrial robots to blood serum diagnostic equipment. In many cases, software technology is being integrated into products by technical staff who understand hardware but are often novices in software development.

The personal computer has been the catalyst for the growth of many software companies. While the software companies of the second era sold hundreds or thousands of copies of their programs, the software companies of the third era sell tens and even hundreds of thousands of copies. Personal computer hardware is rapidly becoming a commodity, while software provides the differentiating characteristic. In fact, as the rate of personal computer sales growth flattened during the mid-1980s, software-product sales continued to grow. Many people in industry and at home spent more money on software than they did to purchase the computer on which the software would run.

The fourth era in computer software is just beginning. Object-oriented technologies (Chapters 8 and 12) are rapidly displacing more conventional software development approaches in many application areas. Authors such as Feigenbaum and McCorduck [FEI83] and Allman [ALL89] predict that "fifth-generation" computers, radically different computing architectures, and their related software will have a profound impact on the balance of political and industrial power throughout the world. Already, "fourth-generation" techniques for software development (discussed later in this chapter) are changing the manner in which some segments of the software community build computer programs. Expert systems and artificial intelligence software has finally moved from the laboratory into practical application for wide-ranging problems in the real world. Artificial neural network software has opened exciting possibilities for pattern recognition and human-like information processing abilities.

As we move into the fourth era, the problems associated with computer software continue to intensify:Hardware sophistication has outpaced our ability to build software to tap hardware's potential.

Our ability to build new programs cannot keep pace with the demand for new programs. Our ability to maintain existing programs is threatened by poor design and inadequate resources.

In response to these problems, software engineering practices--the topic to which this

book is dedicated--are being adopted throughout the industry.

An Industry Perspective

In the early days of computing, computer-based systems were developed using hardware-oriented management. Project managers focused on hardware because it was the single largest budget item for system development. To control hardware costs, managers instituted formal controls and technical standards. They demanded thorough analysis and design before something was built. They measured the process to determine where improvements could be made. Stated simply, they applied the controls, methods, and tools that we recognize as hardware engineering. Sadly, software was often little more than an afterthought.

In the early days, programming was viewed as an "art form." Few formal methods existed and fewer people used them. The programmer often learned his or her craft by trial and error. The jargon and challenges of building computer software created a mystique that few managers cared to penetrate. The software world was virtually undisciplined--and many practitioners of the clay loved it!

Today, the distribution of costs for the development of computer-based systems has changed dramatically. Software, rather than hardware, is often the largest single cost item. For the past decade managers and many technical practitioners have asked the following questions:

Why does it take so long to get programs finished?

Why are costs so high?

Why can't we find all errors before we give the software to our customers?

Why do we have difficulty in measuring progress as software is being developed?

These, and many other’ questions, are a manifestation of the concern about software and the manner in which it is developed--a concern that has tend to the adoption of software engineering practices.

## 中文译文：

# 软件和软件工程

## ——软件的出现及列举

在二十世纪八十年代的前十年开始的时候, 在商业周刊杂志里一个头版故事大声宣扬以下标题：“软件，我们新的驱动力！”软件带来了一个时代------它成为了一个大家关心的主题。在八十年代中期，《财富》杂志的一个封面故事在哀叹：“一个在发展的软件的缺口 ”在这十年结束时,商业周刊杂志的经理被警告只因为关于那句“软件陷阱自动装置或者其他”。在九十年代破晓之初, 在纽约时代杂志上有个特写询问：“我们能信任我们的软件吗?” 并且华尔街时报叙述了一家专业软件公司通过辛苦的努力的头版文章题为“创造新的软件是苦恼的任务。”这些标题和很多其他的类似的，是那种重要的电脑软件的一种新的理解的先兆, 是新理解的作先驱计算机软件的重要机会, 它提供并且形成的危险。 软件现在已经超过了硬件，作为许多计算机成功的电脑基础系统的钥匙。无论计算机被用来经营业务，控制一个产品工程，或使系统运行，软件是区分的因素。信息的完整性和实时性由一个公司的不同的竞争者的不同软件(和相关的数据库) 提供。软件产品的设计和图案的功能来自人类，区分它从只能选其中之一的产品以一个相似的作用来看。那理解力和功能被插入的软件提供，经常区分二种相似的工业或消费品。 这就是有利有弊的软件。

在计算机时代的前三十年期间, 第一位的主要挑战就是发展电脑计算机硬件减少处理和存放数据的费用。在八十年代的十年过程中,微观电子学的发展导致计算能力逐步提高而成本却越来越低。今天,问题不同了，在九十年代第一位的主要挑战是发展电脑基础解决办法的软件执行质量(和减少费用被实施以软件的)为主。

一台 1980 年代主要结构主机的力量是现在一台办公桌上可利用的。现代硬件的令人敬畏的处理和贮藏能力代表计算的潜力。软件是使我们利用和轻拍这潜力的机制。

软件角色的演变

在计算机系统发展的五十年左右的时间里，软件紧密地配合着其发展。更好的硬件性能，更小的尺寸，更少的花费已经促成更好的计算机基础系统。

我们移动真空电子管处理器微观电子学的设备已有能力每秒运行二亿条指令。在流行书籍上的电脑革命，有科学家描绘它为“一场新的工业革命”。有的科学家说微型电子学的发展是人类历史上第三次巨大的变化，有的预言道一场工业革命在向信息社会发展在我们的生活中将有一个意义深远的效果，有的说道电脑控制的信息和知识将成为 21 世纪的焦点力量，有的提出“电子社区”将被网络工作创造以及软件是世界上相互交换知识的钥匙。

在九十年代初，有的科学家描述到一个力量变速器在旧的建筑（政府的、教育的、工业的、经济的、军用的）将一体化作为计算机或者软件导致的知识的民主化。

描述的是在计算机为主的系统应用区域之内软件的演变的时间表。 在早年计算机系统发展期间, 硬件进行了连续变动。那时侯软件作为事后的不被注意想法被许多人观看了。计算机编程是少量系统的方法存在的艺术。软件开发实际上是不被关注的—— 直到日程表滑倒了或费用开始升级。在这个期间, 批取向被多数系统使用了。也有例外的，著名的例外是交互式系统譬如早期的美国航空保留系统和实时针对防御系统。甚至在很大程度上，硬件是执行一个专用执行文件程序的专用物件。

在早年期间, 通用硬件变得普遍。软件，另一方面，是按客户要求设计的为各种应用和有相对地有限地发行。

软件产品(i.e.,节目显现出被卖对一个或更多顾客) 是在联邦税务局初期。多数软件由同样人或组织开发了和最终使用。您写了它, 您得到了它的运行, 并且如果它失败了, 您固定了它。由于工作流动性是降低, 如果遇到问题负责人也能放心。

由于这是个人性化的软件环境, 设计是在一个人的头脑里含蓄过地程执行, 并且文献经常是不存在的。

在早年我们学会了关于计算机为主的系统的实施, 但较少学习关于计算机系统工程。但是, 我们必须承认被开发在这个时代期间的计算机为主的系统是有许多卓著之处的。这些令人倾慕的成就有很多保留在今天的使用中提供继续纠错。

第二个时代计算机系统演变(图 1-1) 跨过了十年即从 60 年代中期对 70 年代晚期。

多元化的程序和多用户系统介绍了人与机器互相作用的新概念。交互式技术打开了应用硬件和软件优雅新水平的一个新世界。实时系统能收集、分析，并且变换数据表多个来源，因此控制过程和生产产品在毫秒而不是分钟。其优点在网上存贮导致了数据库管理系统的第一代。

第二个时代为软件公司出现并且描绘产品软件的用途。软件为普遍发行在一个多重学科的市场上这个目的而被开发了。软件为计算机主机和微型计算机吸引了上百和甚至数以万计的用户。

从此，企业家从产业、政府以及学术界打破了“开发最后软件包”和赢得捆绑金钱的模式。

如同计算机为主的系统的数量在增长一样, 计算机软件图书馆开始扩展。机构内部的发展项目导致了成千上万个程序源语句。

被购买的软件产品从外面增加了成千上万个的新声明。但是犹如一朵黑暗的云彩出现在天际,所有这些程序——所有这些有一些应用是产生的缺点来源状态被查出了被改, 当用户要求修改则改变, 或适应被购买的新硬件。 这些活动集

体叫做软件维护。在软件维护上工作人员花费了很多的功夫，但是软件还是以惊人的比率吸收资源得到资源。

更糟糕的是, 许多程序的个人化的本质使他们潜伏的盟友失去了运转的可靠性。一个“软件危机”在世界上隐约地出现了。

计算机系统演变第三个时代开始在 70 年代中期直到今天。 分配系统—— 多台计算机, 各一致地起作用并且执行通信，很大地增加了计算机为主的系统的复杂。

全球性和地区网络, 高带宽数字通信,已经在软件开发商那里对瞬间数据存取的增长的需求投入了重大需求。

第三个时代为对微处理器的出现和以及其普遍用途, 个人计算机，以及强有力的小规模工作站。

微处理器产生了大多智能产品，比如从汽车到微波炉, 从产业机器人到血液清液诊断设备。在许多情况下, 软件技术是联合了解硬件的产品的技术职员开发的，但经常是新手在开发软件。

个人计算机是许多软件公司成长的催化剂。如果说第二个时代被卖的是上百或数以万计的软件公司他们的程序的拷贝, 第三时代出售就是十和甚而成千上万个的拷贝的软件公司。

当软件提供区分的特征的时候，个人计算机硬件迅速地成为商品。实际上, 作为个人计算机销售成长率被铺平在 80 年代中期期间, 软件产品销售还在继续增长。许多人

在产业和在软件上做购买软件花费的金钱比他们买的在家会运行的计算机更多。

第四个时代对计算机软件来说是正义起点。面向对象的技术(章节 8 和 12) 在许多应用范围迅速地偏移更加常规的软件开发方法。

作者譬如[ FEI83 ] 以及 Allman [ ALL89 ] 预言"五世代" 计算机, 计算的程序根本不同, 并且他们的相关软件在世界各地将对政治和工业力量的平衡方面产生深刻的冲击。

现在，“软件开发的第四代”技术(在这个章节里以后会讨论) 改变方式软件社区修造了计算机程序的一些段。最后，专家系统和人工智能软件从实验室进入了现实世界中的广泛问题，成为其实际应用的软件。人工神经网络软件开辟了扣人心弦的类似人类信息处理能力。

如同我们进入第四个时代的时候，计算机软件的问题继续增强:

问题一： 很多性能优越的硬件已经超过了我们建立软件控制硬件的潜力及能力。

问题二： 我们建立新程序的能力无法与对新程序的需求同步。

问题三： 由于粗劣的设计和不充分的资源，我们维护现有的程序的能力受到了威胁。

这些问题的的回应, 被采取在产业过程中。软件工程实践——这本书的题目是热忱的。

产业透视

在早期计算, 程序员开发了计算机为主的系统使用被安置的硬件管理。项目负责人集中于硬件因为这是为系统开发唯一最大的预算项目。控制硬件费用, 负责人设立了正式控制和技术标准。

在某事被建立了之前，他们会要求详尽的分析和设计。 他们测量过程确定何处能做改进。简单地陈述，他们申请了控制、方法，以及我们认可当硬件工程学的工具。但是很悲哀的是, 软件经常只是少许事后的想法。

在早期, 编程被观看者观看了但是只是作为“艺术形式”。少量正常方法存在了但是也只有很少人使用了他们。

很多程序员经常地经过反复试验大家的编程艺术，然后学会这些复杂的编程工艺。计算机软件专科术语的大厦的杂乱无章和攻击性大厦挑战了少量程序员仔细透视这一奥妙。软件世界是真正的无纪律的—那个时期很多专业人员爱上了软件世界!

今天, 费用的发行被计算机为主的系统的发展显著改变了。 软件, 而不是硬件, 经常是最大的唯一费用项目。过去十年经理和许多技术实习者请求以下问题:

为什么它采取如此长期时间得到被完成的编程?

为什么费用是很高的?

为什么我们不可以发现所有错误在我们把软件给我们的顾客之前?

为什么我们当软件被开发时在测量的进展时有困难?

这些, 并且还有很多其他的问题, 是被程序开发者对软件的关心和对软件工程实践的采纳的关心的一种表现的显示。

## 附录 B 主要参考文献的题录及摘要

1、C#入门经典（第六版）

【书名】C#入门经典（第六版）

【作者】 Karli Watson

【机构】清华大学出版社 2013.12

【关键词】.NET Framework 的含义； CIL 和 JIT

【摘要】C#是 Microsoft 在 2000 年 7 月推出.NET Framework 的第 1 版时提供的一种全新语言。C#的快速流行，使之成为使用.NET Framework 的 Windows 和 Web 开发人员无可争议的语言选择。他们喜欢 C#的一个原因是其派生于 C/C++的简洁语法，这种语法简化了以前困扰一些程序员的问题。尽管做了这些简化，但 C#仍保持了 C++原来的功能，所以现在没有理由不从 C++转向 C#。C#语言并不难，也非常适合于学习基本编程技术。易于学习，再加上.NET Framework 的功能，使 C#成为开始您编程生涯的绝佳方式。

2、C#高级编程(第 9 版)

【书名】C#高级编程(第 9 版)

【作者】Christian Nagel, Jay Glynn, Morgan Skinner

【机构】清华大学出版社 2015.1

【关键词】C#与.NET 的关系；语言的互操作性

【摘要】为了理解.NET 的重要性，就一定要了解过去 20 年来出现的许多 Windows 技术的本质。尽管所有 Windows 操作系统在表面上看来完全不同，但从 Windows 3.1(1993 年引入)到 Windows 8.1 和 Windows Server 2012 R2，在内核上都有相同的 Windows API 用于 Windows 桌面和服务器应用程序。在我们转而使用 Windows 的新版本时，虽然 API 中增加了非常多的新功能，但这是一个演化和扩展 API 的过程，并非替换它。

3、数据库系统概论（第 4 版)

【书名】数据库系统概论（第 4 版)

【作者】王珊,萨师煊

【机构】高等教育出版社 2006.5

【关键词】编程与实现；查询优化和并发控制

【摘要】本书系统全面地阐述数据库系统的基础理论、基本技术和基本方法。全书分为 4 篇 17 章。基础篇包括绪论、关系数据库、关系数据库标准语言 SQL、数据库安全性和数据库完整性，共 5 章；设计与应用开发篇包括关系数据理论、数据库设计和数据库编程，共 3 章；系统篇包括关系查询处理和查询优化、数据库恢复技术、并发控制和数据库管理系统，共 4 章；新技术篇包括数据库技术新发展、分布式数据库系统、对象关系数据库系统、XML 数据库和数据仓库与联机分析处理技术，共 5 章。

4、网页制作与数据库应用

【书名】网页制作与数据库应用

【作者】赵国玲等

【机构】电子工业出版社 2006 年 03 月

【关键词】网络；网页设计；数据库设计；数据存储；JSP

【摘要】书是按照“优化结构、精选内容、突出实用”的指导思想所编写的一本全

面介绍计算机网络与数据库基础知识、应用与操作的教材。全书内容包括 3 个模块化结构。模块 1，网络与网页设计知识与操作；模块 2，数据库应用知识与操作；模块 3，程序设计基础知识与操作。

5、Web 应用开发技术

【书名】Web 应用开发技术

【作者】高文会

【机构】机械工业出版社　2005 年 8 月

【关键词】java；Web；数据库访问技术；网络安全

【摘要】介绍了 Web 原理及应用开发方法，同时深入浅出地介绍了被广泛使用的 Web 应用技术：发技术：JAVAScript 客户端开发技术,JSP/Servlet、AD6、JDBC、JavaBean 等服务器端开发技术，全面系统地展示了如何使用这些技术生成动态页面、操纵数据库、建立 Web 站点。

6、ASP.NET 入门经典(第 9 版)

【书名】ASP.NET 入门经典(第 9 版)

【作者】 William Penberthy

【机构】清华大学出版社　2016 年 11 月

【关键词】Web Forms；ASP.NET MVC；部署

【摘要】在互联网的早期，大部分内容都以静态方式创建和存储。每个 HTTP 请求都请求的是具体的页面或存储内容，响应只会提供该内容。早期的应用程序框架改变了这种模式，允许基于一组特定的标准动态生成内容，并作为请求的一部分发送。于是，内容从数据库和其他来源中建立，成倍增加网络的有效性。

7、UML 建模过程及在需求分析中的应用

【书名】UML 建模过程及在需求分析中的应用

【作者】黄贤英

【机构】机械工业出版社 2001 年 2 月

【关键词】UML；Web；建模

【摘要】UML 是建立系统模型和分析业务处理流程的强有力的工具，从不同的角度描述系统，帮助分析人员弄清楚系统“做什么，谁做，如何做，何时做，以什么顺序做”。描述使用 UML 的需求建模过程，并说明在稽查征费系统需求分析中的应用、获得的经验及注意事项。

8、ASP.NET MVC 5 高级编程(第 5 版)

【书名】ASP.NET MVC 5 高级编程(第 5 版)

【作者】加洛韦

【机构】清华大学出版社　2015 年 2 月

【关键词】TDD；框架；辅助方法

【摘要】 ASP.NETMVC 3 带来了像 Razor 视图引擎这样的新特性，与 NuGet 包管理系统和 jQuery 内置整合来简化 Ajax 开发。ASP.NETMVC 5 继续这一趋势，添加了更新的可视化设计、移动 Web 支持、使用 ASP.NET Web API 的 HTTP 服务、内置支持 OAuth 与流行网站的整合等。这样我们就可以快速地开始使用全功能 Web 应用程序。

9、软件工程

**【书名】**软件工程

**【作者】**程成，陈霞

**【机构】**机械工业出版社　2007 年 4 月

**【关键词】**java；Web；数据库访问技术；网络安全

**【摘要】**全书重新组织为 7 篇，第 1～6 篇涵盖了整个软件开发过程各个阶段的内容，从初始的需求导出到设计和开发，再到软件项目管理。最后一篇论述了一些重要的软件工程的新技术。.本书既有理论高度又有具体操作，非常适合作为高等院校本科生或研究生的教材，对软件工程的管理者和参与者而言亦是必不可少的参考书。

10、构建之法 现代软件工程（第二版）

**【书名】**构建之法 现代软件工程（第二版）

**【作者】**邹欣

**【机构】**人民邮电出版社　2015 年 07 月

**【关键词】**效能分析；技术和流程

**【摘要】**软件工程牵涉的范围很广, 同时也是一般院校的同学反映比较空洞乏味的课程。 但是软件工程的技术对于投身 IT 产业的学生来说是非常重要的。作者邹欣有长达 20 年的一线软件开发经验，他利用业余时间在数所高校进行了长达 6 年的软件工程教学实践，总结出了在 16 周的时间内让同学们通过 “做中学 (Learning By Doing)” 掌握实用的软件工程技术的教学计划，并得到高校师生的积极反馈。在此基础上，作者对软件工程的各个知识点和技能要求进行了系统性整理，形成教材。