

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。汽车配件管理系统帮助企业进行售后服务的配件管理,整合配件信息,统计各种配件业务数据,处理配件出库、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。

本论文研究课题根据一家国有汽车制造企业的实际业务发展需要,以基于工作流的汽车配件管理系统为研究目标,针对国内某汽车厂的实际业务需求进行了认真分析。总结和比较了现有研究成果,以企业配件管理的实际情况为依据,详尽的分析配件管理系统的功能需求,运用了 UML 工具进行系统建模,通过用例模型、流程图、时序图等分析设计手段对系统进行了详细描述。论文对系统的业务功能进行了详细的设计,给出了系统各个模块的类图。论文还对配件管理系统的流程特征、配件管理系统运行时的人机交互以及系统流程的执行过程进行了详细的分析。研究了企业配件管理中的工作流程特征,通过分析流程管理中的活动特点,对系统的流程管理进行了详细的设计,以对系统中业务功能的配置来处理企业配件管理过程中的业务流程重组的问题。并且对系统数据库进行详细的设计,建立了流程管理模块的 E-R 模型,对数据表的逻辑关系和数据库逻辑结构进行了设计。论文的最后对流程模板的定义,系统的整体设计,流程步骤间的消息触发等实现技术进行了讨论。

【关键字】汽车配件管理系统; 工作流; UML; 模型

Abstract

With the rapid development of information technology, information-based management has become an important way to help the corporation coping with challenge, participating in market competition and reducing management cost. The Automobile parts management system assists the corporation to conduct parts management in after-sale service, to integrate parts information and to collect business statistical data of various parts and to deal with complex business on issues, warehousing, inventory and so on. All those make it possible for the corporation to realize the long-range planning, to improve administrators' management skills and professional ability, to supply scientific support for the enterprise development and to achieve efficient management.

This paper study the automobile parts management system based on workflow according to the real business requirements of an automobile manufacture enterprise. It compared and summarizes the current research achievements. This paper detailed analyzes the functional requirements of automobile parts management system according to the real conditions of corporation. It makes use of UML tools to build system modeling and prescribes the system detailed by analysis and design means such as use-case model, flow chart, and timing diagram and so on. This paper designed the system's business functions and the class diagram of every module. It has studied flow characteristics in automobile parts management. By analyzing operating characteristics of workflow in the process, it Detailed design of the workflow management module. It is used to deal with problems on business process restructuring in automobile parts management process by configure the Business functions of the system. It is detailed for the design of database system, database logical structure and logical relationship between data sheets .It also establishes an E-R model for workflow management module. Lastly, this article discusses about implementation technique for definition of the process template, overall design of the system and triggering messages between process steps and the like.

Key Words : Automobile parts management system; Workflow; UML; Model

西南交通大学

学位论文版权使用授权书

本学位论文作者完全了解学校有关保留、使用学位论文的规定，同意学校保留并向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅。本人授权西南交通大学可以将本论文的全部或部分内容编入有关数据库进行检索，可以采用影印、缩印或扫描等复印手段保存和汇编本学位论文。

本学位论文属于

1. 保密□，在 年解密后适用本授权书；
2. 不保密☒，使用本授权书。

（请在以上方框内打“√”）

学位论文作者签名：崔勇杰

日期：2009.6.8

指导老师签名：

日期：

2009.6.8

西南交通大学学位论文创新性声明

本人郑重声明：所呈交的学位论文，是在导师指导下独立进行研究工作所得的成果。除文中已经注明引用的内容外，本论文不包含任何其他个人或集体已经发表或撰写过的研究成果。对本文的研究做出贡献的个人和集体，均已在文中作了明确的说明。本人完全意识到本声明的法律结果由本人承担。

本学位论文的主要创新点如下：

- 1.详细分析了配件管理系统的业务需求和业务流程特征，并根据系统的流程特征提出了系统业务重组的解决方案。
- 2.以工作流的技术实现方法为主线，设计并实现了汽车配件管理系统。
- 3.设计并实现了系统中的工作流任务管理器和流程模板管理器，使之可以对配件管理系统进行流程定义和流程监控。帮助企业实现了业务重组。

学位论文作者签名：

崔家杰

日期

2009.6.8

第一章 绪 论

1.1 问题的提出

我国汽车制造业自本世纪初以来取得了快速的成长,汽车制造商的数量也不断增加,市场的容量还在不断的扩大。加入 WTO 以后,我国的汽车制造业面临更大的竞争,这个竞争不但是生产和销售的竞争,其中至关重要的一部分也是售后服务的竞争。售后服务涵盖了汽车售后保养、配件更换、技术培训、市场等等一系列内容^[1]。汽车制造企业没有售后服务就会失去其存在和发展的基础^[2]。很多汽车制造企业认识到售后服务的重要性,专门成立了三包服务中心,建立售后服务部来提供更好的售后服务,增加企业的竞争力。而汽车配件管理,是汽车售后服务中的一个核心的组成部分。配件管理是指,汽车制造厂在对客户进行售后服务时,对企业的配件库存、配件的销售、回收、进货渠道、发货时间、技术服务等等都要进行统一的管理,在努力实现较少库存甚至零库存的经济模式下,为用户提供及时、稳定、灵活、高效的售后服务。企业售后服务管理部门,对外需要与配件经销商、业务员、配件供应商、售后服务商进行沟通,内部还需要与企业采购部门、财务部门等不同的部门进行沟通。通过内部和外部的统一协作,对用户提供服务。随着企业规模的扩大,业务复杂度的提高,售后服务方式不断的增多,同时伴随竞争的增加,要求售后服务部门响应速度提高,服务成本降低,管理更加规范。而手工、电话、传真等信息传递方式已经无法适应新的售后服务配件管理的需要^[3]。所以,如何利用信息化实现企业信息的协同;改变企业传统作业方式;提高企业管理效率;降低企业售后服务成本;帮助企业提高竞争力。这些都是亟待解决的问题。

汽车配件管理系统,是在信息化背景下,以汽车制造业的业务需求和发展要求为指导,以企业信息化为手段发展起来的一种帮助企业加强管理的信息系统,是汽车制造业发展到一定阶段的产物。它通过对汽车配件进行规范的出入库管理、配件经销商的账目审核、提供各种配件的出入库记录、发货以及销售记录,分析配件销售情况和各个地区的配件库存量、为企业的配件计划制定,杜绝配件流失、降低企业管理成本和售后服务成本,提供了有效的保障。传统的汽车配件管理系统是针对企业一定时期的业务功能需求设计

的,但是由于企业的管理是一个不断进步、不断完善的过程,企业的管理模式本身处于形成、发展阶段,管理的思想、方法、手段是不断变化的,管理方式随着管理思想的变化而变化,而新的管理方式必然引起企业的业务流程重组 BPR (Business Process Reengineering)。所以,随着企业售后服务配件管理方式的不断变化。怎样让配件管理系统适应不断变化的业务过程,对系统开发是一个挑战。

目前,不少大型汽车制造企业选择 ERP (Enterprise Resource Planning) 系统来对企业大部分业务进行集中管理,这种 ERP 系统是一个大而复杂的面向功能的事务处理系统,业务逻辑复杂,包罗万象,将企业内部作为一个大的逻辑流程加以研究,对企业的业务管理进行粗粒度的分析和开发,但是对企业某些专业方面的业务如汽车售后服务配件管理领域缺乏细化的研究。并且传统 ERP 系统对专门的业务流程缺乏有效的控制和管理,使得复杂流程间的资源矛盾和资源冲突依然存在,特别是那些共享的可重用资源。所以,对企业来说,有必要针对企业某些专业领域的业务需求开发适合企业业务管理的信息管理系统。

综上所述,研究符合汽车制造企业售后服务中配件管理需求,如何建立满足企业业务需求,适应企业业务重组需求的轻量级、功能可重构、流程可改变、柔性化、中粒度的企业基于工作流的配件管理系统是这篇论文研究的核心内容。

1.2 国内外研究发展现状

自 1993 年成立了 workflow 管理联盟(Workflow Management Coalition, WPMC)之后^[4],该组织颁布了一系列 workflow 产品标准,包括 workflow 参考模型、workflow 术语表、workflow 管理系统各部分间接口规格、workflow 产品的互操作性标准等。这些举措加速了 workflow 技术的商品化。近年来,workflow 技术得到了长足的发展。现在,许多公司都基于这些标准推出了自己的工作流产品。workflow 产品主要分为两大类:

(1) 基础的工作流系统:提供引擎、设计器、相关接口等。应用系统的开发商可以在此基础上开发具有 workflow 管理功能的应用软件。

(2) 应用 workflow 技术:专门为应用所定制的应用级软件系统^[5],包括内置比较完整的工作流功能,这种系统是直接面向最终用户的流程化应用。同时系统中还往往根据应用的需要,集成了其他功能。

本文研究的中心是基于工作流的汽车配件管理系统,属于应用 workflow 技术。所以下面就配件管理的方式和 workflow 技术的研究两个方面分别对其国内外发展的现状分别加以分析。

1.2.1 国外研究发展现状

国外对售后服务配件管理方面研究的比较早,信息化程度比较高,管理思想比较先进。它们的售后服务管理有“4S”的售后服务和“连锁经营”两种方式。所谓“4S”是指车辆销售、售后服务、零配件供应和信息反馈“四位一体”的汽车经营方式。而所谓“连锁式经营”是指集成了汽配供应、汽车维修、快速养护为一体的综合性服务。这种模式相对前一种来说利用整合汽车配件的资源打破了纵向垄断。这种方式对配件管理来说,是一种更新的方式,可以将资源更合理的整合。国外汽车制造企业很早就采用了基于业务协同的 workflow 管理系统来对企业进行管理。许多国外大型厂商都建立了基于 workflow 技术的企业级协同管理系统,他们大部分通过一个平台将整车厂、配件供应商、配件经销商、售后服务以至于采购等一系列业务整合在一起。

在 workflow 技术的研究角度来看,国外起步的较早。20 世纪 80 年代中期,国外就已经有功能相对单一的商品化 workflow 系统问世,这可以看作早期 workflow 管理软件的雏形。21 世纪以来,由于对 workflow 技术的迫切需要,其研究日益受到人们的重视,国外许多大学和企事业研究机构都开展了很多研究项目,并且取得了众多的研究成果,但是其大部分都是基于 workflow 引擎的 workflow 系统。典型产品如 Action Technologies Inc. 的 Action Workflow、IBM 的 Flow Mark。国外比较知名的 workflow 管理系统供应商有:Ultimus、IDS Scheer、K2。国外 workflow 产品共同的特点是拥有核心产权,技术先进,价格昂贵,业务流程复杂、管理方法多样,但具体的实现技术没有公开。

1.2.2 国内研究发展现状

随着我国经济的迅速发展,我国汽车制造业的管理理念也在迅速发生着变化,许多国有或合资大型汽车制造企业在“走出去”和“引进来”的同时也引进了国外大型企业的先进管理理念和售后服务配件管理系统,利用这些技术和手段对售后服务的配件部分业务进行有效管理,如:“一汽丰田”汽车在售后服务中的最大优势就是大型配件管理系统的应用,他们遵循的原则是“卖一买一”,每日订货和发货都保持频繁、周期性、小批量的特点,实

现了“零库存”。从而大大节约了资金，降低了企业管理和服务的成本。但是另一方面，对于很多我国中小型制造企业来说，他们的业务较灵活，管理思想不成熟，信息化程度也比较低^[9]。他们的管理模式相对简单，既没有实力进行“4S”式售后服务，也没有经济能力引入大型的工作流管理系统对企业各个方面进行管理，甚至很多小型企业还停留在原始的纸质化流程阶段。

国内对工作流相关技术的研究工作开展得比较晚。产品相对较少，其主要原因是因为一方面对产品的重视程度不够，另一方面是国内对系统软件的开发能力有限^[7]。国内比较知名的工作流软件系统供应商有：炎黄盈动、金蝶、神州数码等。典型产品有神州数码的工作流管理软件 EasyFlow，就是以工作流技术为核心的全面的企业办公自动化(OA)产品。其产品最大的特色是采用基于域的联邦系统框架，对分布式管理、运行支持较好，而且也是目前国内为数不多的可以支持“仿真”的工作流系统^[9]。工作流技术在国内刚刚起步，其研究主要集中在如何柔性建模上，而针对配件管理领域的业务需求，以及企业业务流程的执行方面研究的非常少^[9]。

从工作流技术和售后服务配件管理系统相结合的角度来看，用户对于企业管理软件的需求点，已经转向工作流管理^[10]。用户应用管理软件的目的，已经由沟通转为“使团队协作工作节点有条不紊推进”。而目前的大部分企业的配件管理系统工作流全部固定，启用前需要对用户的工作流执行逻辑进行严格设定，启用后各项事务按设定好的流程自动化处理，并且不能修改。对于现代企业的柔性化管理需求，这类协同软件满足起来比较困难^[11]。本文利用工作流技术来解决企业业务流程重组的难题，开发基于工作流的汽车配件管理系统是目前解决企业信息化和业务灵活性需求的优选方案。

1.3 课题来源和研究内容

本文作者参与了汽车产业链协同商务平台以及依托平台建设的国内某汽车制造企业售后服务配件管理系统的设计以及开发工作。参与了系统的需求，设计以及最终的编码实现过程。并且依托此系统进行了工作流模块的研究与设计，设计并实现了基于工作流的配件管理系统，达到了良好的效果，使系统即符合了企业配件管理方面的需要，又满足了企业进行业务重组的需要。

论文的主要研究和开发工作可以分为以下几个部分：

(1) 分析和研究了系统的业务目标以及业务过程。

(2) 对配件管理系统的业务进行具体的需求分析，建立系统的功能模型，流程模型，用例图，以满足汽车配件管理商业目标及要求。

(3) 以研究适应企业业务变化，业务重组为依据，在需求分析的基础上，对系统进行了总体设计，对系统的功能模块和数据库进行了详细的设计说明。

(4) 研究 workflow 技术并且使之成为系统开发基础，设计了 workflow 管理模块，使开发系统满足业务流程重组的需要。

(5) 研究系统的开发框架，采用 .NET 作为系统的主要开发框架，最后对系统的部分功能实现进行了说明。

1.4 论文组织结构

本文共分为五章，每章内容具体如下：

第一章：阐述了当前国内外对基于 workflow 的配件管理系统的研究现状，说明了本论文课题的来源、研究内容和组织结构。

第二章：详细的分析了企业的业务需求，并且通过 UML 对系统进行建模，在分析业务需求的过程中对主要业务模块进行了分析，并且给出了业务流程图、用例图、和时序图。

第三章：首先给出了系统的解决方案，以 workflow 设计为主线按照系统的模块划分给出了系统的详细功能设计。

第四章：提出了系统流程管理的解决方案，分析了配件管理系统的流程特征，针对配件管理系统对流程管理模块进行设计，最后设计了系统的数据库。

第五章：详细描述了系统所用的开发环境和开发工具，并且给出了系统的部分功能模块的界面实现和代码实现。

第二章 汽车配件管理系统需求分析

2.1 企业现状及业务目标

2.1.1 企业现状

山东某汽车制造厂(以下简称 WZ)是我国中型汽车制造企业。拥有员工 7000 多人,总资产近 10 亿元。主导产品有三轮汽车、轻型汽车、客车、皮卡车、农业机械、电动车、汽车配件等多个系列 1000 多个品种,其系列产品畅销全国,并已出口 20 多个国家和地区。

WZ 集团的配件管理业务由三包中心负责,三包服务中心是 WZ 集团销售公司下属的一个独立核算单位。管理着遍布全国各地 700 多家配件经销商的配件销售、500 多个三包服务点的三包配件管理、300 多家供应商的三包旧件索赔。平均每天需要处理 200 多张“配件需求计划订单”,每个月需要处理几万张“三包服务信息反馈单”。涉及的配件种类近万种;其配件管理业务包括配件销售、新件业务、旧件业务。其业务流程历经制定订单、审批订单、开具发票、销售出库、销售发货等过程,就像一条连续链条,各个过程环环相扣,形成一个有序的流程。在这个流程中,涉及到配件经销商、业务员、中心站、三包服务商等角色和部门。配件管理既有集销售服务于一体的服务模式,又有逐渐增多的特约服务站管理模式,业务管理方式比较灵活。通过调研发现,敏锐扑捉市场动态,根据市场需求进行灵活的生产经营是该汽车公司满足市场需要,发展制胜的秘诀。由于公司业务量较大,生产的车型繁多,配件种类、数量巨大,负责售后服务配件管理的部门业务管理工作相互交叉,给公司的生产管理、零配件销售、三包服务都带来了很大的困难。而目前厂内配件管理过程中还有很多需要手工操作的流程,如在配件销售中有很多信息需要和厂外业务员、配件经销商进行交互信息交互,配件销售订单提交、配件发票查询、库存信息查询等都需要通过电话、传真等形式来完成,在这些手工操作过程中会导致信息传递的不及时和遗漏。这些都成为制约企业快速发展的一个瓶颈。

2.1.2 业务目标

汽车零配件管理在汽车制造厂的售后服务中占有举足轻重的作用,如果

无法对零配件进行有效、准确、及时的管理，就会造成企业成本上升，效率低下，降低售后服务的作用，从而影响企业的生产和销售。汽车制造厂售后服务配件管理涉及到配件销售、新件业务、三包旧件业务三个部分，业务过程比较复杂。厂外人员需要及时与厂内系统进行协同，以保证订单的及时有效。厂内配件具有种类繁多、数量大、占有金额大的特点，必须要求配件管理的规范，同时保证配件账目的清晰准确。旧件回收中必须保证旧件回收的有效性，回收过程的完整性。在二级库配件缺失的情况下，要及时进行二级仓库之间配件的调度。所以，通过调研分析，可以总结企业业务目标如下：

- (1) 能及时获得配件销售订单订购相关信息和三包服务配件消耗信息，保证订单管理的及时性和有效性。
- (2) 实现新件入库、配件销售、出库管理、发货等业务的规范性，保证账目清晰和符合规范，有效预防账务违制。
- (3) 严格按照旧件回收标准，保证三包服务旧件回收、理赔业务的完整性。
- (4) 三包中心库配件调度上可以灵活方便的解决问题，保证业务调度的灵活性。

2.2 汽车配件管理业务流程分析

通过对企业的调研以及三包服务中心组织机构的分析，可以发现配件管理最为基本的任务是在配件销售、新件管理、旧件管理方面，另外还有车辆档案、综合查询等方面的功能。

本文研究方向为基于工作流的配件管理系统设计，所以着重对这三个方面的业务流程进行分析。用统一模型语言(UML)进行业务建模，以用例图和业务流程图进行描述。下面将从配件销售、新件管理、旧件管理等方面的业务入手，进行详细的系统业务需求分析。

2.2.1 配件销售管理

随着企业生产及销售规模的不断扩大，企业配件销售的业务内容逐渐从单一向多方面发展，同时业务对象也越来越多。目前，配件销售涉及的业务对象有二级库、业务员、配件经销商经销商、售后服务商、配件代理商等；涉及的业务内容有销售订单、销售发票明细、销售出库、销售发货、收货信息管理等。配件销售是一个大的业务流程，各个表单之间都有着业务联系，

配件销售要由配件经销商、配件代理商、客户提交销售订单给三包中心库审批，三包中心库根据相关条件进行审批，只有审批过的订单才能安排发货等活动。这里对其业务活动进行统一的分析。

● 配件销售业务分析

配件销售流程图如下图 2-1，以下是对配件销售的业务分析：

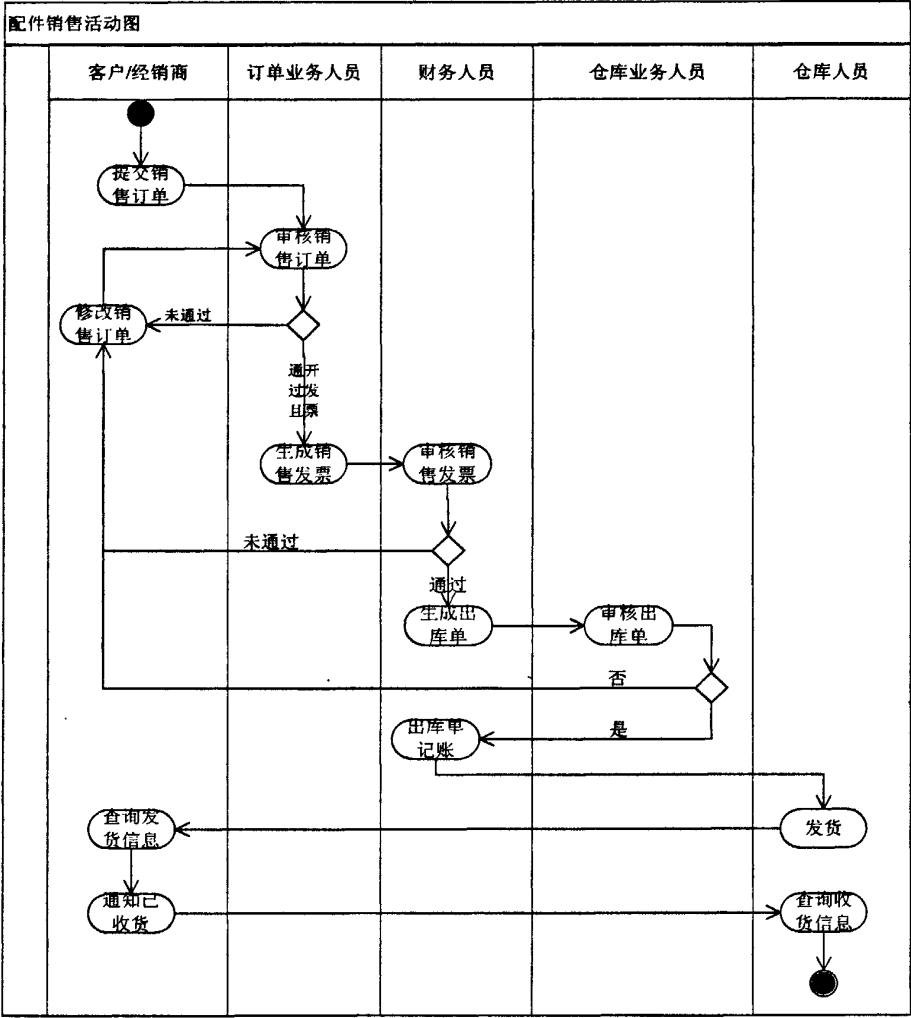


图 2-1 配件销售流程图

- (1) 配件经销商，代理商或其他客户根据销售情况提交销售订单。
- (2) 厂内订单业务人员对订单进行审核，检查订单内容是否合理和规范，如：订购车辆的头灯，如果订单未说明是左灯还是右灯，厂内业务人员需要与销售订单下单人员进行联系核实，并通知他修改订单。
- (3) 订单业务人员审核并通过符合规范，配件信息清晰的订单。并且由

审核后的销售订单开销售发票给财务人员审核。

(4) 财务人员审核销售发票, 查看配件经销商科目余额, 如果发现经销商余额不足, 则通知经销商汇款。如果经销商已经汇款, 则重新查询经销商账目, 账目余额足够则审核通过销售发票, 并生成销售出库单。若经销商不再付款, 则审核不通过, 同时通知经销商修改订单减少订购数量或撤销订单。

(5) 仓库业务人员收到销售出库单, 并且根据销售出库单上配件内容检查库存, 对出库单进行审核。库存配件有一个警戒水平, 低于库存警戒水平但足够发货则先记账、发货并安排采购入库, 如果仓库零件高于警戒水平, 可以进行销售。则审核通过, 财务人员进行记账, 同时对配件经销商的账目进行扣款, 如果库存数量已经达不到销售需要, 并且没有办法按时采购交货, 则审核不通过, 并联系经销商修改订单, 先安排销售一部分配件。

(6) 出库单记账后, 根据出库记账单, 仓库人员对销售的零配件进行出库和安排司机带运, 进行配件发货。

(7) 客户或配件经销商可以从中心库查询订单审核情况和发货信息。

(8) 三包服务中心通知客户或配件经销商已经发货, 当客户或经销商收到货后, 向三包中心联系告知到货情况。

通过对上面配件销售流程图的分析, 我们可以发现, 配件销售模块中业务流程相对比较复杂, 各个业务活动面向的对象比较明确, 业务活动相对比较独立。企业的管理方式在不断的发展变革中, 业务管理方法也会发生变化, 由以上模块我们可以发现, 在配件销售过程中, 从订单的提交到发货一共经历订单提交、订单审核、生成销售发票、审核销售发票、生成出库单、审核出库单、出库单记账、发货 8 个业务活动。这些业务活动可以随着业务的管理方式发生变化, 在不违背业务逻辑规律的情况下, 从一种流程变为另一种流程。如: 以上流程可以变为先订单提交、订单审核、生成出库单、生成发票、最后发货的流程。

● 配件销售用例模型

通过以上分析, 可以发现, 配件销售的业务流程参与者主要有客户/经销商, 财务人员, 销售订单业务人员, 仓库业务人员。用例则有提交销售订单、修改销售订单、审核销售订单、开销售发票、审核销售发票等。通过相似行为分析和扩展行为分析的用例优化, 可以得知, 审核销售订单后客户/经销商可以查看订单审核情况; 生成销售发票后客户可以查看销售发票; 仓库业务员审核出库单需要查看库存信息; 财务人员审核销售发票需要查看往来账目款项。配件销售用例图如下图 2-2 所示:

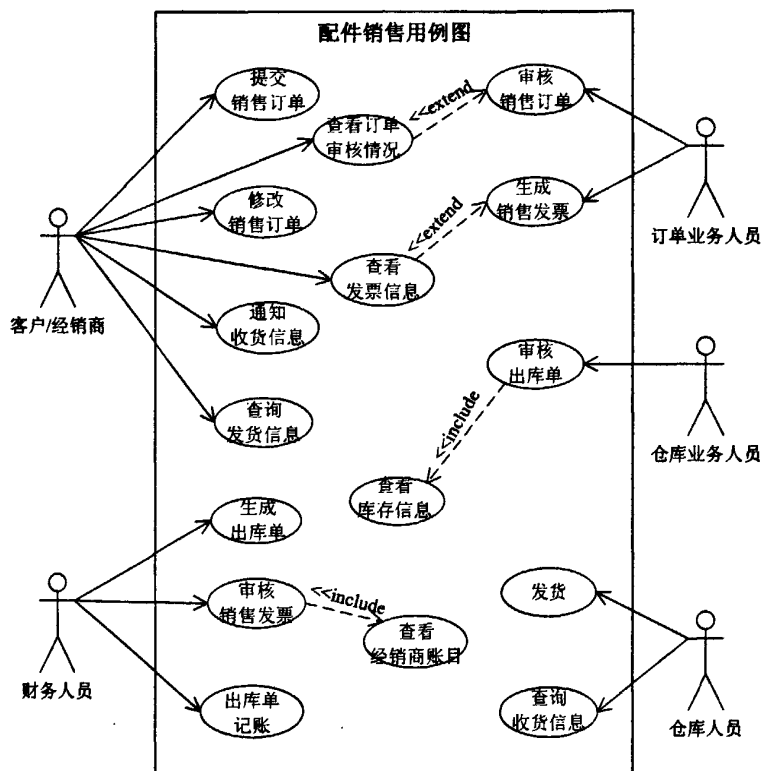


图 2-2 配件销售用例图

2.2.2 新件管理

新件管理是配件管理系统比较核心的一个部分,因为新件管理包括制造厂所有的新件的业务,其业务穿插在制造厂的从采购到生产销售的各个领域。主要业务涉及对仓库的所有新件入库,新件盘点,新件出库的管理。新件入库及出库的业务比较灵活且相对独立,下面逐一对其进行功能建模并进行用例分析,因为新旧件仓库盘点流程基本一致,所以新件盘点作为配件盘点的一部分在后面集中讨论。

● 新件入库业务分析

所谓新件入库也就是针对新件进入仓库的业务流程。一般新件入库是指外购零件入库,也就是汽车制造厂从零部件供应商那里采购零配件进行的入库行为。通过对厂内的业务进行了解,厂内新件入库业务还包括另外两部分,即盘盈入库和其他入库。其中盘盈入库和盘点单密切相关,且由盘点单生成,这里暂不做讨论,后面在分析盘点业务的时候将详细分析。其他入库是指仓库的新件不只是外购一种方式进入,也许会有其他特殊情况需要的而发生的

新件入库行为，如：新件临时借出,需要在一定时间内归还入库,这种新件归还的方式就属于其他入库。其他入库也是新件入库，所以其入库流程与用例与外购入库完全相同。

如下图 2-3，新件入库调动流程包括的步骤有：

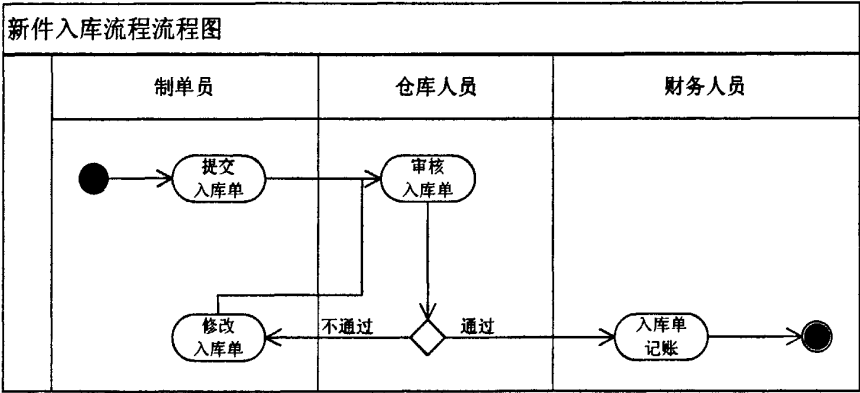


图 2-3 新件入库流程图

- (1) 制单员根据入库货物制作入库单表，提交入库单。
- (2) 仓库人员对入库单进行审核，审核入库单是否和出库配件相符。如果无误则通过审核，否则审核不通过，通知入库单录入人员修改入库单。
- (3) 财务人员将审核后的入库单进行记账，库房人员将配件入库。已记账后表明配件入库流程结束。

● 新件出库业务分析

新件出库也就是针对新件离开仓库的业务流程。其中新件出库中的大部分业务是指零配件的销售出库，其业务流程在配件销售模块已经进行过描述，这里不再重复。另外一部分是盘亏毁损的出库，这个作为盘点的一部分将在后面进行详细的分析和建模。还有一部分属于其他出库，其它出库也指临时出库(不包括销售,销毁,放弃)的新件,有时需要在一定时间内归还入库，同样入库的新件对应的入库方式为前面提到的新件归还。这三种方式都属于新件出库的行为，只是对其业务处理有所不同，如在销售出库中，出库单是要凭销售发票开具，而其他出库中，需要人工直接填写出库单。下面讨论的主要是对新件其他出库业务进行分析和建模。

新件出库流程图如下图 2-4：

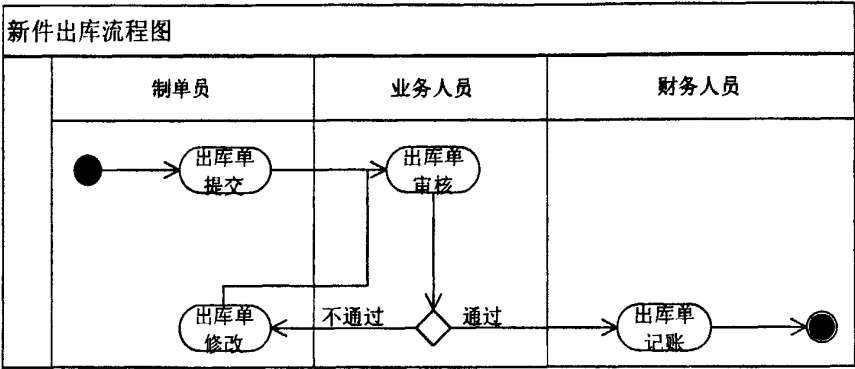


图 2-4 新件出库流程图

新件出库的工作流程：

- (1) 内部人员根据出库配件定制出库单表，提交出库单。
- (2) 仓库人员对出库单进行审核，若单据无误，则审核通过。反之，审核未通过，通知出库单录入人员进行修改。
- (3) 财务人员对审核后的出库单记账，对应库房库存作相应减少。已记账后表明配件出库流程结束。

从上面业务分析来看，新件的入库和出库都包括单据的提交、审核、记账，当厂内的业务发生变化时，也有流程变化或将审核和记账业务流程简化的需求。

● 新件管理用例模型

新件入库和新件出库中，参与者主要有制单员、业务人员、财务人员，涉及到的用例主要有入库单提交、入库单修改、入库单审核、出库单提交、出库单修改、出库单审核。通过相似行为分析和扩展行为分析的用例优化，可以得知，制单员制单的时候，需要查看配件基础信息；业务人员在进行审核出库单的时候，要查看库存信息检查库存量，检查配件是否可以出库。可以画出新件管理用例模型图如下图 2-5 所示：

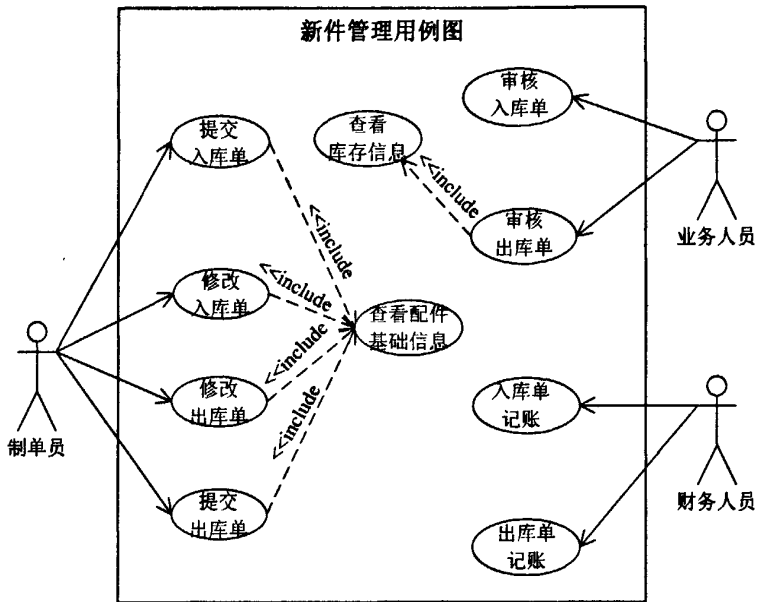


图 2-5 新件管理用例图

2.2.3 旧件管理

旧件管理业务是以旧件库为中心，涉及的业务对象有供应商、业务员、服务站等。涉及的业务内容为旧件收、发、存管理。其中在旧件仓库管理也分为三个方面，旧件入库，旧件出库，盘点。

● 旧件管理业务分析

旧件入库主要用来处理业务员，配件经销商返回的旧件入库业务，可以手工录入入库单信息，也可以通过平台传回来的返回三包旧件明细单自动生成旧件入库单。暂存入库主要是用来处理业务单位旧件返厂后，没有通过旧件三包入库审核的旧件暂存入库业务。

旧件出库主要用来处理旧件退回配件供应商的旧件出库业务。旧件出库还包括暂存出库，暂存出库主要用于处理暂存旧件的出库业务，这两种业务可以统一为旧件的出库业务。

旧件发票与新件发票不同，旧件发票是针对内部的业务人员录入的旧件入库单、旧件销毁单、旧件直接返配套厂单和旧件出库单开具的，旧件发票是给配件供应商开具的，并由供应商返回旧件款项。而新件发票是在配件销售中，给业务员或配件经销商开具。如下图 2-6，旧件管理主要流程包括的步骤有：

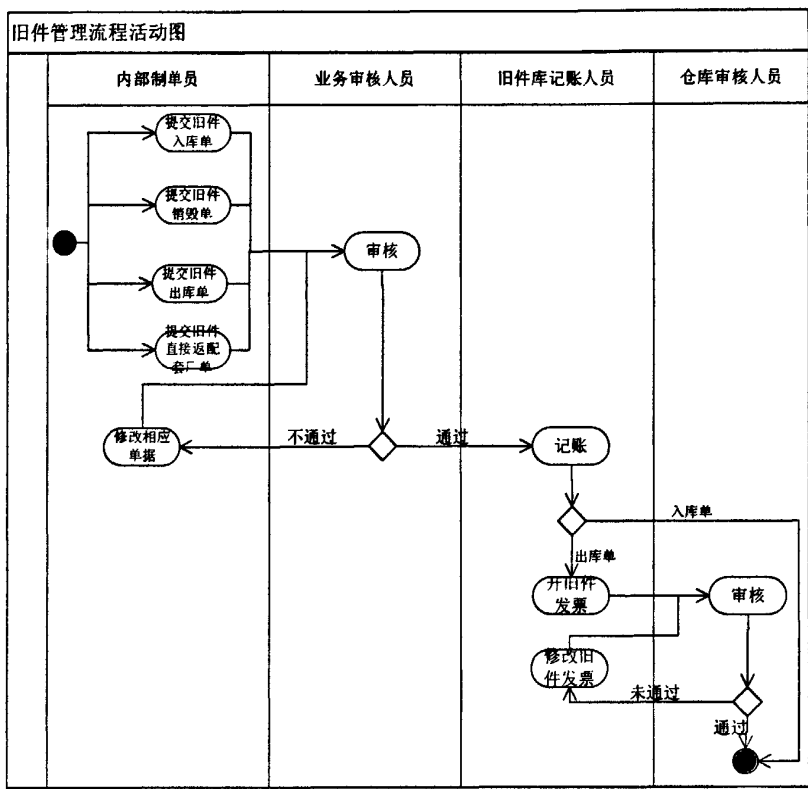


图 2-6 旧件管理流程图

- (1) 旧件入库单用来处理业务员，配件经销商，返回的旧件入库业务。
 - (2) 旧件销毁单用以处理业务员或经销商就地销毁或处理旧件的情况。
 - (3) 旧件直接返配套厂单处理把旧件直接返给配件供应商的情况。
 - (4) 旧件出库单用来处理旧件退回配件供应商的旧件出库业务。
 - (5) 仓库审核人员对这些单据予以审核，符合实际情况则通过审核，反之通知其修改相应的单据。
 - (6) 财务人员审核后的单据进行记账。
 - (7) 如果是出库单，则根据记账后的单据可以给供应商开旧件发票明细，业务审核人员进行对照审核。
 - (8) 审核通过则业务流程结束，审核不通过则通知修改旧件发票。
- 通过对旧件管理业务的分析，可以发现涉及旧件管理的单据都包含有审核和记账的业务。随着企业业务调整，某些单据中审核和记账业务有简化、合并、或变更的需求。

● 旧件管理用例模型

旧件管理中,参与者主要有制单员、仓库审核人员、财务人员、业务审核人员,涉及到的用例主要有提交旧件入库单、提交旧件销毁单、提交旧件出库单、提交旧件直接返配套厂单、仓库人员审核、财务人员记账,业务人员审核。通过相似行为分析和扩展行为分析的用例优化,可以得知,仓库审核人员进行审核的时候,可以查看出库、入库等单据信息。同样,业务审核人员通过查看发票的信息对旧件发票明细进行审核,修改相应单据的时候也要先对单据进行查看。旧件管理的用例模型图如下图 2-7 所示。

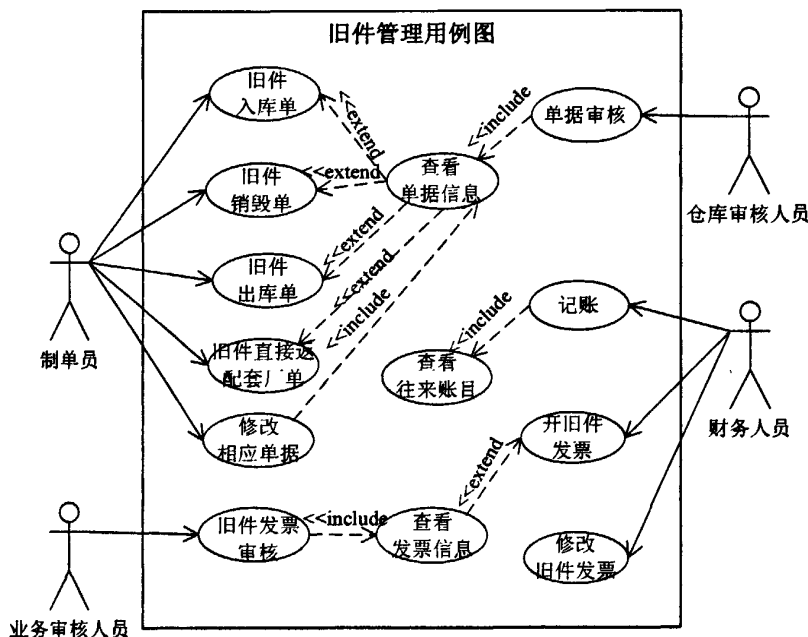


图 2-7 旧件管理用例图

2.2.4 配件盘点

配件盘点的业务是围绕着新旧件仓库的入库和出库来进行的,有配件的入库,就有配件的出库。到了每个月的月结期间,新件仓库和旧件仓库需要对仓库里的零件进行盘点。保持账目的收支平衡。在对仓库的零件进行盘点以后,可以发现零件是盘亏还是盘盈,假如仓库里的配件因某种原因而毁损,那就需要开盘亏毁损单。假如仓库零件比实际账目要多,那就需要开盘盈入库单,盘盈入库是一种发现盘盈之后(即仓库配件实际数目要比账目数目多的一种情况),这时应该补充账目,也就是用盘盈入库来补充。盘盈入库也是新旧件入库单的一种,其业务流程和配件入库一样,这里就不在赘述,下

面对盘点进行用例分析及建模。

● 盘点业务分析

如下图 2-8，盘点流程包括的步骤有：

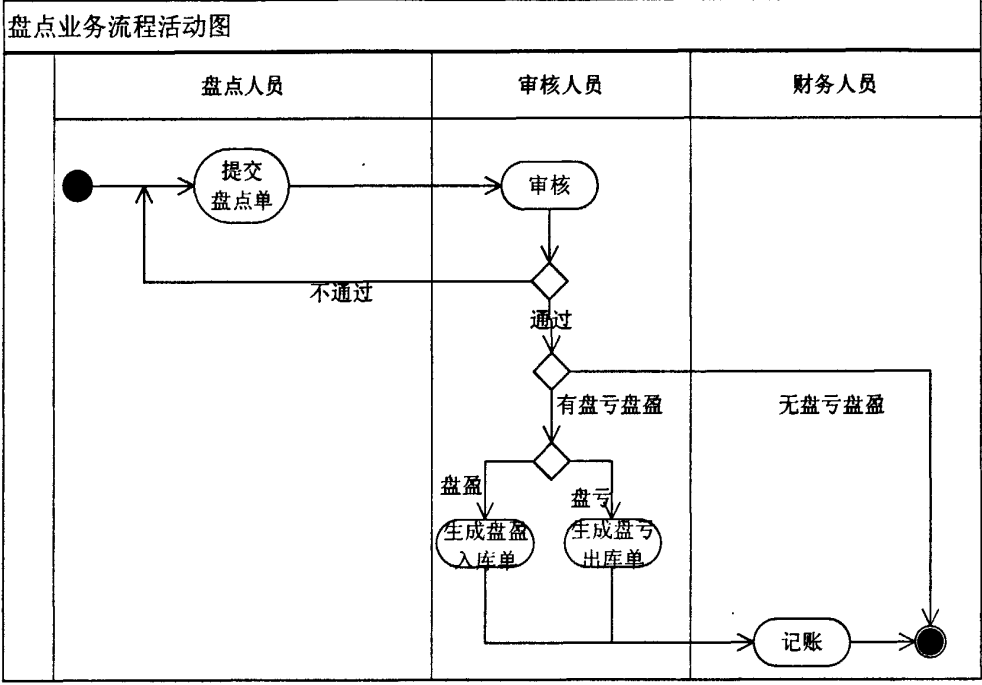


图 2-8 盘点业务流程图

- (1) 盘点人员选择仓库进行盘点工作，对某类的配件进行盘点。
- (2) 盘点人员得到库房的实际库存明细后，对应盘查实际库存数量，并据实填写盘点单。
- (3) 审核人员对盘点单进行审核，审核不通过则重新进行盘点，审核通过则根据盘点单开盘亏、盘盈单。如果盘点和库存一致则工作流程结束。
- (4) 财务人员将盘盈入库单和盘亏出库单记账，对应库房或业务员库存作相应增加或减少。

● 盘点用例模型

盘点业务中，参与者主要有盘点人员、审核人员、财务人员，涉及到的用例主要有盘点、开盘亏入库单、开盘盈出库单、审核、记账。通过相似行为分析和扩展行为分析的用例优化，可以得知，审核人员在进行审核的时候，可以进行查看库存信息的查询，而财务人员进行记账的时候，可以查看往来账目。进行盘点的时候，可以进行盘点单的查询，库存信息的查询。所以盘点用例模型图如下图 2-9 所示。

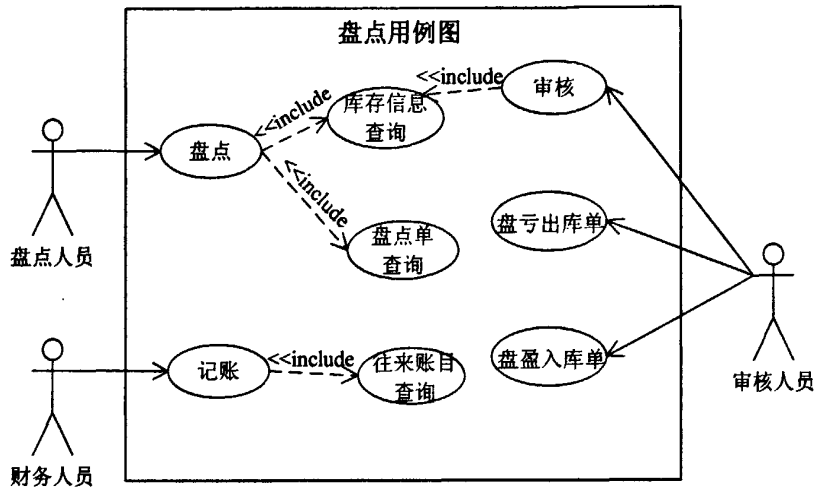


图 2-9 盘点用例图

2.2.5 非功能需求

(1) 协同性需求

企业三包服务用户分布非常分散，所以配件经销商、二级中心库和业务员也遍布全国各地，在对汽车零配件进行三包服务和配件销售的时候针对的区域非常广，这就要求厂外的业务员、配件经销商和其他用户在下销售订单或者在对旧件返厂进行三包服务的时候要与厂内进行及时的业务协同。所以系统首先应该满足企业业务协同性的需求，满足厂外人员与厂内进行实时信息交互，数据传递的业务要求。

(2) 灵活性需求

企业现行的业务流程仅符合目前的企业业务管理方法。随着企业管理思想的进步，管理制度发生变化，管理的方法以及手段发生改变，企业业务流程不可避免的要进行调整。当企业工作流程发生重组，配件管理系统要可以进行简单的配置以适应业务流程变化的需要，在不需要修改代码调整应用程序业务逻辑的情况下，可以进行简单的企业业务流程重组。满足工作流程的灵活性。

第三章 系统的设计

3.1 系统总体设计

3.1.1 系统协同性设计方案

上一章分析了配件管理系统的业务需求以及业务流程，并对它们的业务流程加以提取、抽象。从他们的实际情况不难发现，企业的业务员、二级中心库、配件经销商、售后服务商遍布各地。他们和制造厂本身之间的距离很远。他们的信息交流也是远程的，必须要解决他们的协同性问题。这就决定了他们可以有两种方式可以实现系统的解决方案。一、厂内像国外的大型厂商一样，建立自己的商业信息化平台，业务员、中心库、配件经销商通过自主建设的平台与厂内系统进行交互。完成售后服务配件管理流程。这就是说汽车制造厂要自主开发专门的针对其企业内部业务与外部信息发布相集成的平台，其平台应该满足厂内配件管理的业务需求，而不仅仅是一个起宣传作用的网站。这种方案的缺点是实施成本高、培训难度大、人力耗费大。首先，企业需要有自己的系统开发团队，购置开发软件和硬件设施，有专门的人员进行系统的维护，成本非常高昂。其次，各家制造厂对自己的业务团队进行培训，不仅是厂内系统的人员还有厂外和系统流程相关的人员。最后，他们的平台要部署到运营商(中国电信，中国网通等)托管的服务器上，出了问题要专人进行解决，同时也需要一笔专门的费用。二、利用通用平台的方式实现，也就是租用第三方 ASP^[12]平台，利用互联网来解决，ASP 厂商对企业所需的服务进行集成，发布到 ASP 平台上去，提供给企业所需要的应用服务。业务员、中心库人员、配件经销商登录平台，进行订单的业务操作，厂内系统与平台进行数据交换获得信息。通过采用这种企业外包应用服务的租赁模式^[13]。节省了大量的人力、设备、资金、避免了重复的劳动，资源的浪费，同时企业的服务得到了专业的管理和维护。由于第三方平台是根据本行业具体业务设计实现的一个专业型服务平台，讨论和研究具有行业针对性的业务需求，依托专业的开发团队进行开发，在行业内有通用性，专业性。所以这种方案设计专业，管理规范。同时，相对于企业来说成本较低，企业只需要支付一定的租赁费用，省去了研发、系统维护等一系列事务，为企业大大降低了信息化成本。企

业只需要考虑自身的核心业务上，而无须花费精力在系统的集成、开发和维护上面。

综上两种解决方案的比较，相对于中小企业来说，无疑第二种方案具有明显的优势。因此本文将采用第三方 ASP 平台来实现系统的部署。汽车产业链协同商务平台由四川省制造业信息化研究院组织开发^[14]。汽车产业链协同商务平台是在研究了汽车制造厂的需求，以汽车制造企业为核心，专门组织研发的针对于汽车行业的数据共享中心。它可以支持多个企业与业务员、配件经销商、三包服务商、配件供应商等远程关系人与制造厂之间的业务协作，从而建立上下游关系，形成一个产业链。平台以数据交换的形式与厂内系统进行信息交换，配件经销商、业务员通过平台录入和查询厂内信息。这种方式不但大大的降低了企业费用，而且增强了企业间的协作，增加了企业业务运作的效率。系统的协同性设计方案如下图 3-1 所示：

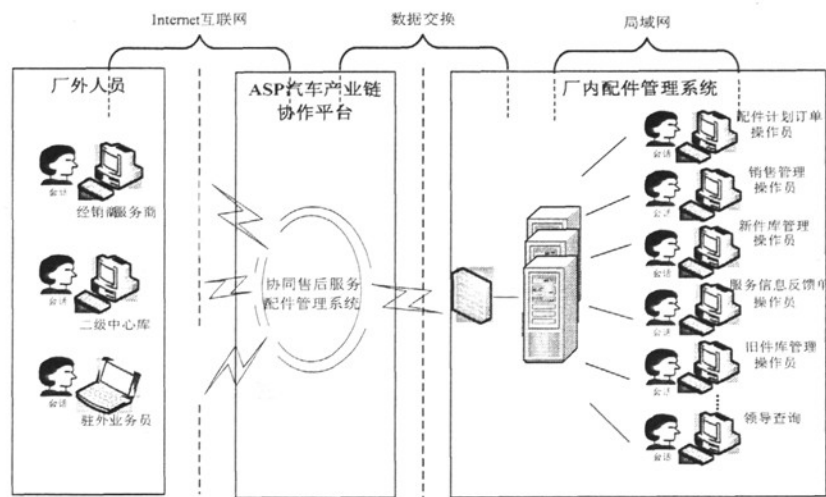


图 3-1 系统协同性设计方案图

系统采用 B/S（ASP 平台）加 C/S（厂内配件管理系统）的混合结构，通过对平台与厂内系统实时的进行数据交换，很好的解决了系统的协同性问题。采用这种结构的好处是，对于厂外配件经销商、业务员等人员来说只通过 ASP 平台进行订单的录入和信息查询。而对于厂内人员来说，他们只面向配件管理系统，这样对于用户来说，他们之间就是对象与对象间的互操作，中间的平台和系统对他们来说是透明的，他们所考虑的就是对业务的处理，而不必花精力在完成业务操作需要的方式上面。本系统以这种整体的设计方式来满足企业厂内与厂外业务协同的需求。

3.1.2 系统流程管理设计方案

从上面的非功能需求分析中可以得知, 企业用户有业务流程重组(BPR)的需要。即在用户管理方法发生变化时, 由于企业管理制度发生变化从而导致企业业务流程需要进行相应变化时, 通过对系统简单的配置, 系统中的工作流程可以重新排列来满足企业的业务需要。通过上一章的需求分析, 可以发现在配件管理系统中很多业务模块的业务过程重复性比较强, 比如订单销售, 新件入库、出库等。它们都包括提交、审核、记账等类似的功能。在每个业务过程中, 面对的参与对象比较明确。业务流程可以实现定义, 经过一个完整的流程才能完成所有工作。这就为将业务操作行为作为一个活动进行业务流程定制提供了可能。

目前的工作流系统, 大部分都是基于文档和消息传递的办公自动化工作流系统。因为最早工作流的目的就是为办公自动化而服务的, 而当前的大多数企业办公自动化软件, 在企业的信息系统中起的作用非常小, 他们只是实现企业的最基本的办公作用, 比如说日常的办公管理、文档传递和一般的限于角色的审核流程。这些都只是企业信息化管理中最基本的部分。

配件管理系统各个模块不同于办公自动化系统。它的重点在于研究企业的业务, 包括对企业业务数据的收集、处理和分析。从这个角度来看, 这是他们的差异。但是, 这并不能说他们之间完全没有联系, 根据企业业务需求的变化, 如果可以将办公自动化的流程定制与流程重组融合到配件管理系统中, 将工作流技术引入到配件管理系统中并且可以让企业内部的数据按照需要进行流动, 完成配件管理功能上的业务重新定制以及各种形式表单的定制需求。无疑将大大的增加企业的效率, 降低企业的管理成本。

从企业的业务流程分析可以发现, 配件管理系统的流程活动相对固定, 企业流程是严格按照某个步骤执行下去的, 串行化程度较高。企业管理方式一旦变更, 企业启用一个新的业务流程来管理业务, 其管理的方式方法都会持续较久的一段时间。而各个业务活动相对比较固定, 活动的特征比较突出, 这就为将这些活动(Activity)提取出来, 用代码的形式将活动加以封装提供了方便。

传统上工作流管理系统的实现经常使用运行在 BPEL 引擎环境下的 BPEL(Business Process Execution Language)语言来实现。BPEL 作为一种形式规约语言^[15], 主要是为了实现系统的自动化业务流程。BPEL 利用将流程步骤写入 XML 文档中, 并且将这些流程用标准化的交互方式

在 WEB 服务之间进行组织,最终让这些流程能够在任何一个符合 BPEL 规范的运行环境上执行。一个 BPEL 流程一般会创建一个 BPEL 流程实例, BPEL 就是定义这个 BPEL 流程如何和外部活动、过程组件或 Web Service 进行交互的^[10]。

本系统实现业务流程控制采用了 BPEL 的技术思想,通过将定义的活动步骤写入 XML 文档中,并将业务活动进行连接来实现业务流程的调用。系统对各个业务活动的连接只定义所要调用的接口,这个接口是一个抽象的活动接口,在流程实例运行时对这个接口进行绑定,这样使接口和实现之间达到了一种比较松散耦合的效果,实现了流程的可重用性。简单来说,本系统工作流管理模块设计的思想就是通过内建的流程逻辑判断类来完成对各个业务活动接口的调用,内部工作流管理模块提供了按一定次序顺序集成各种接口的功能,其本质就是利用内部工作流管理控制器定义的活动次序将过程组件“串联起来”,而“串联起来”的意思就是让这些通过接口联系起来的过程组件产生一种动态的交互。所谓动态交互,举个例子来说:将一个变量的值赋给另外一个变量,这种形式就是变量之间的动态交互。

综上所述,本系统是一个企业管理信息系统(MIS),采用 BPEL 工作流技术思想建立轻量级工作流引擎,在企业内部系统上采用工作流的设计思想,开发一个采用轻量级消息传递和业务活动串行化连接技术的基于工作流的配件管理系统。

3.1.3 系统整体功能规划

本文研究的系统是基于工作流的汽车配件管理系统,目的是在实现制造厂配件管理业务功能的基础上可以对业务流程进行灵活的配置,实现功能的重构。所以系统在功能上划分为两部分,一部分是配件管理,一部分为流程管理。流程管理中为了让一个系统在流程处理上可以满足不同步骤的流程定义的需求,就需要建立相应的流程模板,随着企业业务的划分,根据不同的流程可以定制成不同的模板。然后根据建立好的模板进行流程实例调用,企业进行业务重组只需要调用相应的流程实例。这样根据管理过程中使用过的流程实例特征,企业在业务重组的过程中也可以不断的根据业务调整抽取管理流程的共性与特性,相应的使企业的业务管理更加规范。例如:在企业配件销售过程中,总的流程活动相对比较固定,通过预先在模板中对业务活动的重新组合,从销售订单的提交一直到配件的发货,配置这个流程里的中

开发票、出库、以及各种单据的审核和记账等活动流程,从而产生一个新的业务流程。并在用户角色权限管理中定义该步骤的执行角色,

当定制好模板后,当企业因业务需要对模板进行调用,操作员根据业务流程以及业务类型选择模板,调用模板产生流程实例。这样通过调用流程实例中里预先定制的活动的方式,根据活动跳转条件,在一个步骤完成之后可以自动触发下一步。由此可知,本系统流程管理需要包括 workflow 模板管理、工作流任务管理器两大模块,流程模板管理的功能包括流程定义、流程权限分配,对定义好的流程进行编译,对流程步骤中的角色控制划分等功能;工作流任务管理器包括流程查询和流程控制、流程删除三大功能。为了便于管理员在定义完流程之后需要对业务活动的操作角色进行分配,流程管理模块还包括了用户权限管理,通过基于角色的权限分配,用户根据自己的权限可以对系统中的各项业务功能进行具体的操作,完成在业务流程中的具体工作。以上功能将在后面的设计部分进行详细阐述。

由前面系统协同性设计方案得知,为满足厂内用户与厂外用户的业务协同,系统必须解决业务的协同性问题。本系统与 ASP 平台进行交互采用了数据交换来解决这个问题,数据交换类型是业务驱动的,即在有业务需要的时候,由厂内人员对数据交换中要交换的数据进行配置,选择要交换的内容,进行数据交换,用这种方式保证了厂外人员和厂内人员业务数据的协同性。由于本文研究重点在于 workflow 技术在汽车配件管理系统中的应用,这里对系统的数据交换不做深入研究和设计,这里只对本系统数据交换采用的思想和技术方案做简单介绍:

本系统数据交换采用基于 XML 的 WebService 数据交换技术。WebService 本身是平台上发布的一种组件,也可以看作一种应用程序,它对对象的封装性和集成性都很好。系统数据交换采用的协议是 SOAP(Simple Object Access Protocol)。SOAP 作为一个不依赖传输协议的网络层表示协议,可以以对象的形式在应用程序之间交换数据^[17]。SOAP 使用 XML 来格式化消息,用 HTTP 来承载信息。对服务的请求是通过发送 SOAP 消息实现的。企业必须通过调用 ASP 平台发布的 WebService 才能进行数据传输的操作,ASP 的服务器端一直处于一种被动的等待状态,有数据交换需要时,由企业请求服务器端发起请求,调用 WebService 进行数据传输,周期性的对数据进行上传、下载、转换处理、加密解密的操作,与 ASP 平台进行数据交互;服务器端部署在 ASP 平台一端,主要功能包括数据压缩组件、数据处理加密组件以及数据访问组件。根据以上分析,数据交换服务器

部署在 ASP 平台一端，而请求服务器集成在企业内部系统的一端。厂内系统进行数据交换，首先读取系统中 DataExchange.config.xml 配置文件，获得要进行交换的表名、字段名、字段数据类型、控制列、状态、远程方法名等信息。接着初始化数据交换控制器，初始化数据对象。接着进行数据上传或下载，如果是“上传”则根据定义生成 SQL 语句，查询数据库，提取数据信息。由于数据量可能会很大，所以将数据进行数据格式转换，并用二进制序列化后并进行压缩。然后向 ASP 服务器发送请求连接，连成功则用 SOAP 封装、加密并用 Http 协议进行传输。ASP 平台接收到消息后将消息进行解密并对数据反二进制序列化，根据系统调用的 WebService 方法，平台对数据库中存储过程进行调用，将数据更新进数据库中并更新传输状态为“已上传”，数据交换结束。

配件管理系统的业务模块分为配件销售、新件管理、旧件管理三大模块，每个模块包含多个子模块，配件销售应该包括销售订单、销售发票、销售出库单等等一系列单据的制单、审核、记账等功能。新件管理和旧件管理包括新件和旧件的入库、出库、盘点等业务。本文后面将对其进行详细的设计。综上所述，通过对系统各项功能的简单分析，可以得出系统的整体功能详细划分如下图 3-2：

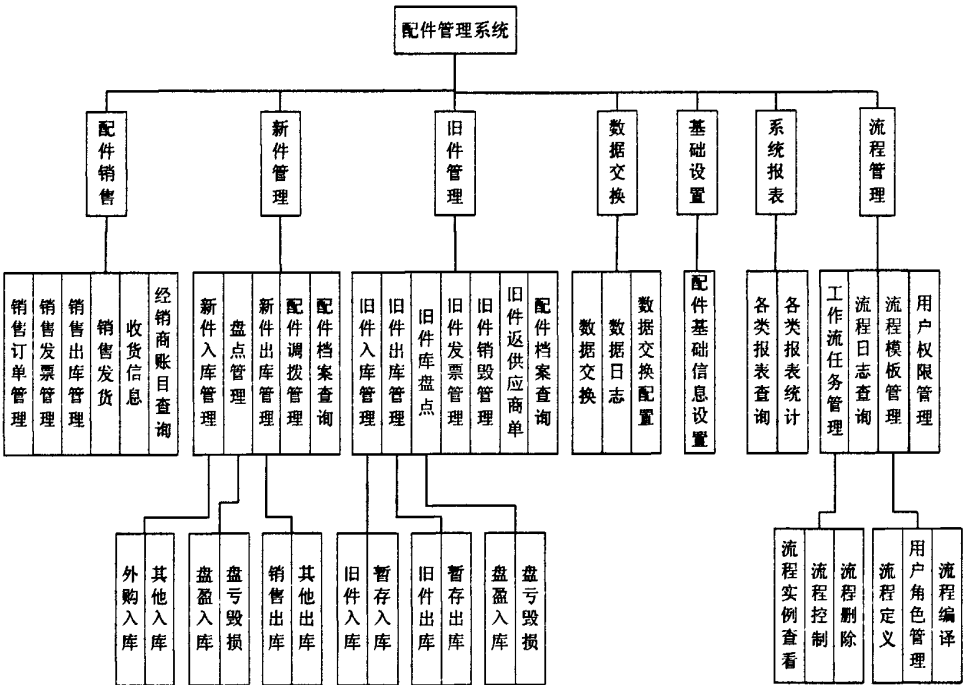


图 3-2 系统整体功能规划图

3.1.4 系统的分层结构

系统整体设计上采用了基于.NET 的分层结构系统。应用系统由四个部分组成，层次结构划分成用户表示层、业务逻辑层、数据访问层、公共数据层四个部分。下图 3-3 为配件管理系统四层结构图：

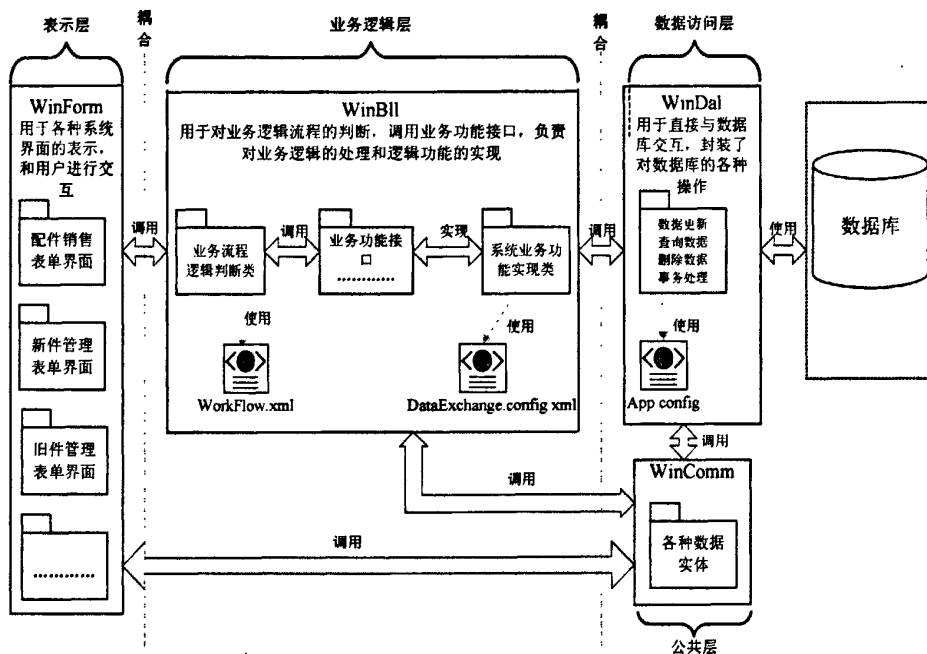


图 3-3 配件管理系统四层结构图

表示层 (WinForm) 主要用于用户数据录入和信息的显示。这一层不负责数据的处理，而是和逻辑层进行交互。为了增强系统的健壮性，在表示层的用户输入框体内要对用户的输入用正则表达式加以验证。

业务逻辑层 (WinBll) 与表示层和数据访问层完全分离，它负责对表示层传递下来的数据进行业务处理，它的业务逻辑被多种客户机共享^[18]，通过业务逻辑层与表示层的分离，使开发人员可以专注于分析和设计核心的业务逻辑，快速建立起系统的核心业务功能框架。其中逻辑判断类通过使用 Workflow.xml 负责工作流程的跳转和判断，调用业务功能接口实现业务操作。业务实现类中数据交换类将调用 DataExchange.Config.xml 中的字段以及方法进行数据交换。

数据访问层 (WinDal) 为业务逻辑层提供数据支持，通过 ADO.NET^[19] 访问数据库，执行业务逻辑层对数据的请求。数据访问层包括连接管理、事务管理、存取管理等操作。数据访问层中的数据连接方法可以调用

App.config 中数据库字符串来进行数据库访问。

公共层 (WinComm) 是一个数据封装层, 主要用于封装各类数据实体和公共基本方法。如: 用户登录信息实体、全局静态常量 (用于保存存储过程名以备调用)、公用基本方法 (转化字符串、转化日期格式等)。这一层特别之处在于可以被其他三层调用, WinForm 层可以调用存储在这里的用户信息等数据实体进行验证等操作; WinBill 层可以调用封装在这一层中的数据进行业务逻辑操作; WinDal 层可以调用这一层中保存存储过程名的静态常量进行对数据库中存储过程的调用。这样做更有利于实现系统的松耦合, 如: 系统需要替换存储过程, 只用在这一层中保存存储过程的类中找到相应变量即可, 避免了对所有数据访问层中类的存储过程进行查找替换。

3.2 系统业务功能详细设计

3.2.1 配件销售管理设计

配件销售管理主要负责制造厂对于配件的销售。它涉及到的功能包括订单的提交, 订单的审核, 发票的开具, 发票的审核、记账, 出库单的生成, 出库单的审核等多种业务类型。配件销售分为销售订单、销售发票、销售出库、销售发货、发货信息、报表查询 6 个子功能模块, 下面对这 6 个子模块进行详细设计:

(1) 销售订单: 用于对系统中配件销售订单的管理, 主要功能有新增、保存、删除、审核、打印、刷新等, 配件订单的一部分为 ASP 平台与厂内系统进行数据交换取得, 一部分为业务员回厂带回的销售订单, 所以需要有信息的录入和提交。点击客户名称下拉菜单选择客户, 会自动出现负责该地区业务的业务员姓名、电话等信息, 操作员输入交货地点、客户联系人、交货地点等信息。点击下方表格中配件编号下拉菜单控件, 出现如配件名称、规格型号、单位、单价等配件各种信息, 选择相应配件, 输入配件数量。通过点击表单上面保存、审核、删除等按钮进行相应操作。

(2) 销售发票: 在配件销售过程利用这个模块为配件经销商开具发票, 同时根据销售发票开出库单。业务人员可以根据销售订单以及订单编号直接生成销售发票。源单号由下拉菜单直接查询, 防止了输入的错误, 销售发票的编号是系统自动生成的流水号, 操作员在开票日期控件选择日期, 生成的销售发票中配件的金额可以自动统计。点击保存可以提交销售发票, 上级主管可以对销售发票进行审核。审核后的发票可以打印和开具出库单。

(3) 销售出库: 用于对配件的出库进行管理, 可以根据销售发票或者销

售订单开具出库单,并且由出库人员根据销售出库单的详情进行配件的出库和发货。系统可以根据销售发票或销售订单开具销售出库单(根据定制好的工作流程判断)。操作员选择源单类型为销售订单或者销售发票,点击单头的源单编号下拉菜单选择,系统自动生成销售出库单信息,操作员选择制单日期、发货仓库、收货单位,点击保存提交。审核后的销售出库单进行记账,对配件经销商扣款。可以进行发货。

(4) 销售发货:主要用于对配件的发货,发货人员可以根据表中的单据信息进行发货,根据销售发货界面中出库日期、客户名称、订单编号、发票编号、出库单编号、配件名称等信息,选择左方复选框进行销售发货。发货单可以导出为 Excel 文档作为发货凭证和留底。

(5) 收货信息查询:通过与平台的数据交换可以得知各个配件经销商和收货部门的收货情况,收货情况可以导出为 Excel 表格形式保存。

(6) 报表查询:可以对销售订单、配件发运安排、销售发票、销售出库单、发货信息、退货统计、经销商账目、销售收入汇总、销售收入明细、客户销售情况汇总等信息进行查询。

根据功能设计,结合本系统的分层结构,可以按层次抽取出以下类,UI(界面)层:销售订单,销售发票明细,销售出库单,发货。BLL(业务逻辑层):销售订单业务类,销售发票业务类,销售出库业务类,流程逻辑判断类。业务层中的将审核、记账等功能封装成活动预留出接口待调用,系统进行流程定义的时候将对这些接口进行重新绑定,用以创建新的业务流程。DAL(数据访问层):数据库操作基类,销售订单数据访问类,销售发票数据访问类,发货数据访问类,销售出库数据访问类。COMM(公共层):数据实体,基类,页面基类,权限管理。

配件销售模块类关系图如下图 3-4 所示:

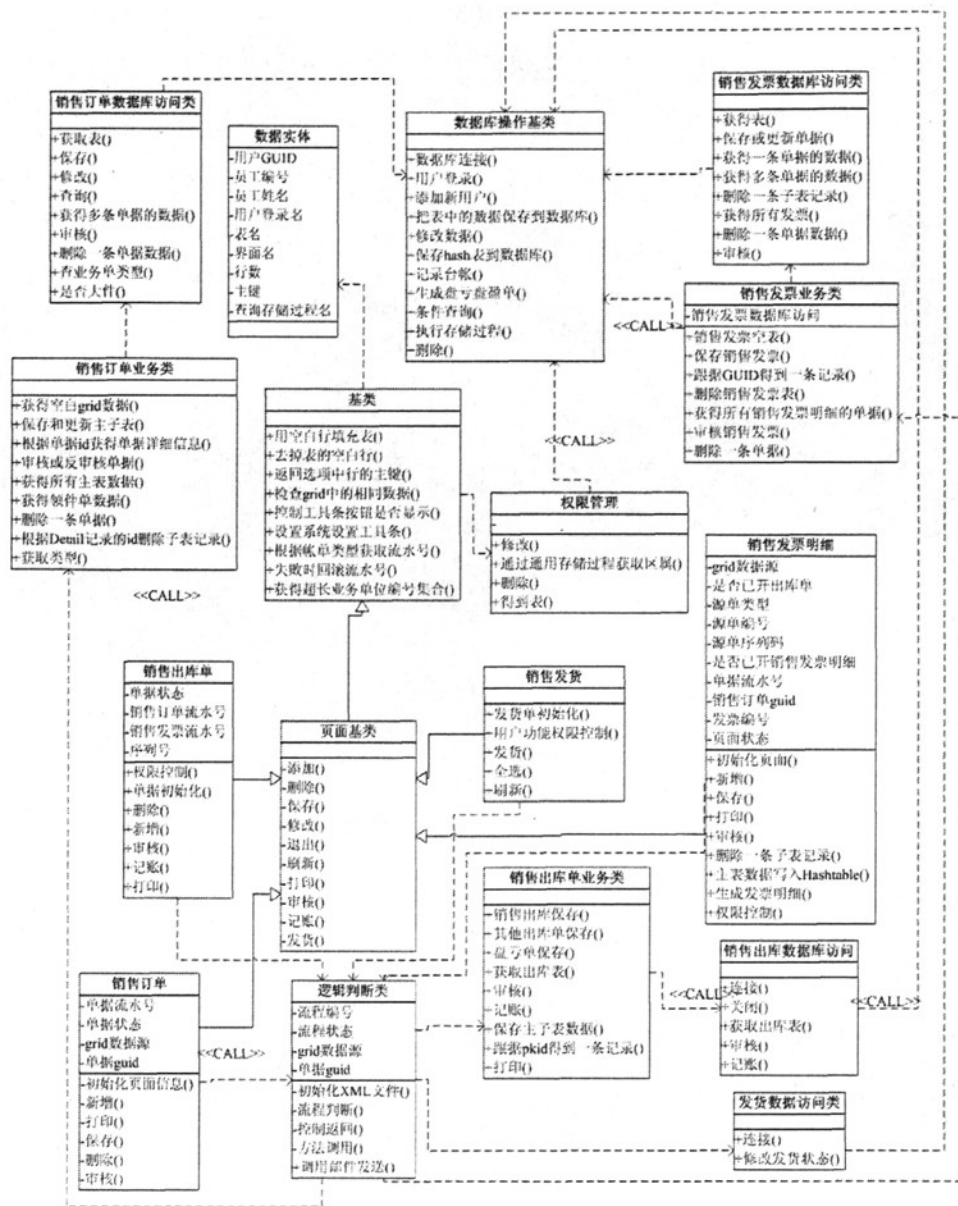


图 3-4 配件销售管理类图

由于考虑到本系统流程的配置，所以在设计中，业务层业务活动功能方法要实现最大程度的解耦。通过调用这些业务活动方法，进行有条件判断，实现系统的工作流程变化。下面结合具体业务分析，以销售订单审核为例，做详细的分析设计：

销售订单类继承了页面基类，页面基类定义了添加、删除、保存、审核等空事件方法体，销售订单类对其需要的添加、删除、保存、审核等功能进

行事件方法体的重写。通过这种方式，在对页面基类的按钮事件进行控制访问中，就可以实现对各个表单的按钮显示方法的控制。另一方面，各个表单界面初始化都要调用逻辑判断类，逻辑判断类调用 XML 中系统的流程配置，并对系统业务功能进行判断，调用各个独立业务活动方法，实现流程控制。

配件销售时序图如下图 3-5 所示：

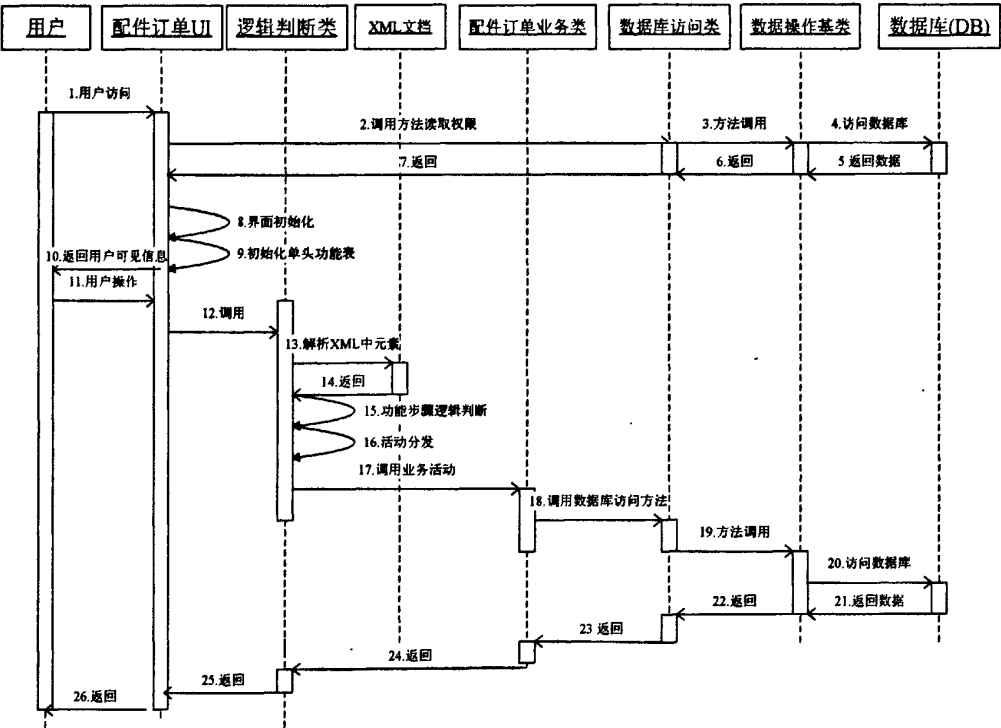


图 3-5 配件销售时序图

配件销售的具体过程：用户打开配件订单界面，配件订单从数据库提取用户权限，初始化菜单功能。用户根据功能对配件销售进行操作，调用逻辑判断类，如果是第一次访问则逻辑判断类读取 XML 中数据到 Dataset 中，并对流程步骤进行判断，进行业务活动步骤的分发，然后信息依次向业务层，数据访问层传递，最后对数据库进行相应的操作。然后数据依层次返回传递，最后通过订单界面展示给用户。

3. 2. 2 新件管理设计

新件管理主要用于处理业务员三包配件库、二级中心库、厂内中心库的出入库业务。利用外购入库、其他入库、盘盈入库、盘亏毁损、销售出库、

其他出库、调拨、盘点等功能来处理厂内中心库的进销存业务；利用盘盈入库、销售出库、盘亏毁损、调拨、盘点等功能来处理二级中心库的出入库业务；利用销售出库、借件来处理业务员的三包配件出入库业务。新件管理分为外购入库、盘盈入库、其它入库、调拨单、盘点单、盘亏毁损、销售出库、报表查询 10 个业务功能，通过业务功能的相似性分析，下面对这 10 个业务功能分成 5 部分进行详细设计：

(1) 入库：入库分为外购入库、盘盈入库、其他入库。主要用来对采购来的配件进行入库的操作，外购入库处理新件的采购业务，盘盈入库负责盘点过程中，实际库存大于盘点库存时，为保存账目收支平衡的一种入库手段。其他入库与其他出库相对应，用于配件借出归还等特殊业务入库的操作。在功能上，操作员选择供货单位、新件仓库、制单日期，并通过选择下方表单中配件编号的下拉菜单选择配件，输入相应数量，进行配件入库，入库单均需要审核和记账功能，记账后的入库单可以打印存档。

(2) 出库：出库分为销售出库、盘亏毁损、其它出库。销售出库在配件销售中已经设计过，这里不再赘述，盘亏毁损用于仓库配件发生毁损时，为保证配件账目收支平衡而采取的一种出库手段，其他出库是一种配件的临时出库行为，他们的功能特点和销售出库一致。

(3) 盘点：盘点用于在每个月结账时对仓库中配件的数量进行账目核对，仓库人员输入配件实际数量，核算员选择仓库，自动会生成盘点单，填写保管员姓名，并进行审核，审核过后的单据可以自动生成盘亏和盘盈单。生成的盘亏盘盈单通过记账，完成本次的盘点操作。

(4) 调拨单：调拨单用于厂外的二级中心库之间的零件调拨之用。在进行配件调拨时，相关业务人员通过选择配件，填写配件数量，选择调出仓库、调入仓库、制单日期，完成保存、审核、记账的业务操作。

(5) 报表查询：报表查询的功能主要有入库单查询、出库单查询、调拨单查询、盘点单查询、中心库/二级中心库库存查询、中心库/二级中心库月结报表查询、中心库/二级库缺货告警等查询功能。

结合功能设计，可以将新件管理模块抽取出以下图 3-6 新件管理模块类图。限于篇幅，新件管理的类图仅画出 UI 层中具有业务操作功能的单据类图及其基本功能方法，其相应的业务逻辑层和数据访问层不再赘述。从下面类图可以看出，新件模块分为盘点单，新件入库，外购入库，盘盈入库，借件单，调拨单，销售出库单，其他出库单，盘亏出库单。从功能上来说，所有出库单和入库单都有审核和记账功能，而盘点单可以生成盘亏出库单和盘盈入库单。其审核和记账功能应该可以予以变动，在厂内业务进行重组的时候，出库单、入库单的提交、审核、记账等功能应该可以需要进行添加删除，并对功能的操作划分角色进行控制。

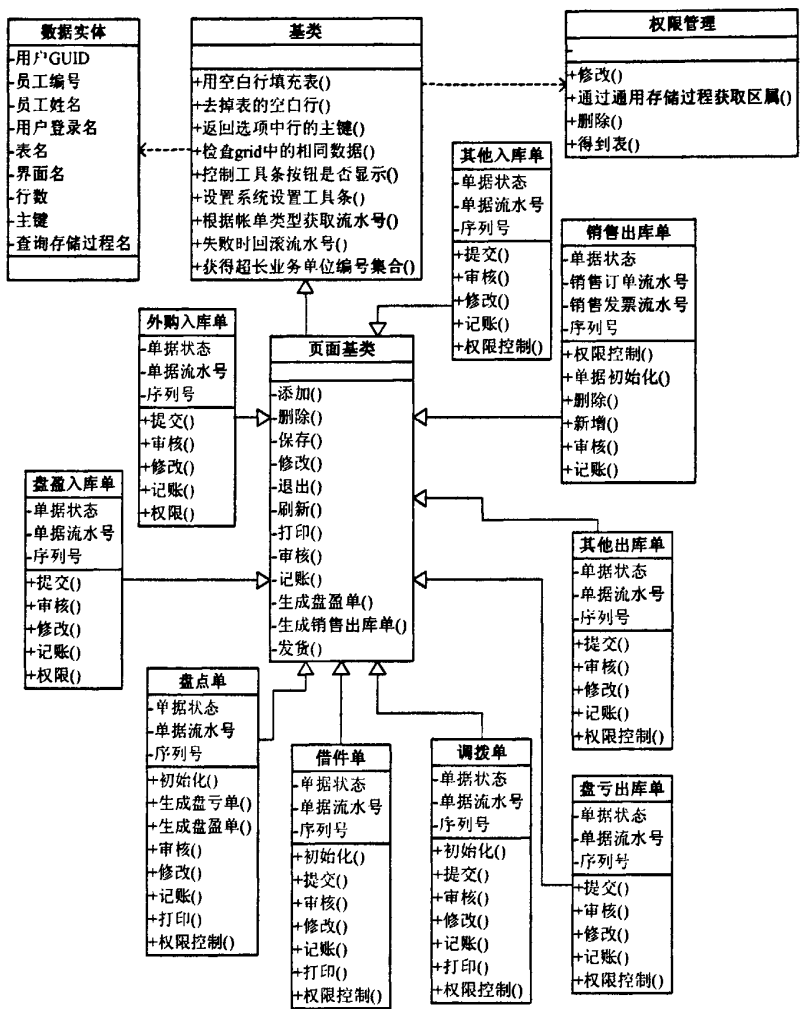


图 3-6 新件管理模块类图

3.2.3 旧件管理设计

旧件管理功能用于业务员、配件经销商（没有业务员、自己做三包业务的）、特约服务站的旧件实物帐和金额帐管理，厂内旧件库的进出存管理，供应商的旧件出库管理、三包服务费管理、供应商质量反馈信息管理。旧件管理分为旧件入库、旧件出库、旧件盘点、旧件发票、旧件销毁、直接返供应商下账单、暂存入库、盘盈入库、盘亏出库、盘点、报表查询 10 个模块，通过业务功能的相似性分析，下面将对这 10 个模块分成 7 个部分进行详细设计：

(1) 入库管理：入库管理分为旧件入库、盘盈入库、暂存入库。旧件入库是指三包旧件的回收返厂进入旧件库；盘盈入库是在盘点过程中，当实际库存大于盘点库存时，为保存账目收支平衡的一种入库手段；暂存入库指旧件的暂时存放。旧件库操作员选择旧件仓库，点击下方旧件表格中下拉列表选择旧件编号，相应会显现出配件名称、规格型号、单位、单价、金额信息。操作员输入数量信息，保存入库单，账务人员利用单头相关功能进行账务的审核、记账、和查询操作。

(2) 直接返供应商下账单：用于将旧件返给供应商时开具的账单，根据配件编号，同样根据表格中下拉菜单选择配件信息，选择单位名称、供应商名称、制单日期，填写备注，保存单据。保存后的单据由相关部门审核记账。

(3) 旧件发票：旧件发票是将返厂旧件退给旧件供应商时开给供应商的发票。旧件发票可以由旧件入库、旧件出库、旧件销毁、直接返供应商下账单生成，操作员选定源单类型下拉菜单中一项，相应源单号下拉列表中出现符合源单类型的源单编号，操作员选择源单编号，可以自动生成相应的旧件发票。相关人员对旧件发票进行审核。

(4) 出库管理：出库管理包括旧件出库、盘亏毁损、暂存出库。旧件出库指旧件返供应商的出库，盘亏毁损用于仓库配件发生毁损时，为保证配件账目收支平衡而采取的一种出库手段，暂存入库是相对于暂存入库的出库行为。出库单都有审核和记账功能，操作员根据出库配件的信息，在出库单中录入相应信息，进行出库管理。

(5) 旧件销毁：用于对仓库内不能返供应商的销毁处理，操作员选择配件信息，单位名称，销毁时间，输入旧件残值金额，通过点击单头的审核和记账功能完成旧件销毁的工作。

(6) 盘点：旧件盘点和新件盘点功能一样，具有对仓库内零件的盘点功能，可以生成盘亏盘盈单，这里不再赘述。

(7) 报表查询：包括对以上所有单据的查询，并且利用“业务单位旧件实时库存”、“业务单位旧件发票明细账”等报表功能查看旧件实物帐和金额帐。

根据以上功能设计，可以抽取出旧件管理模块类图如下图 3-7 旧件管理类图：旧件管理模块主功能表单部分分为旧件入库、盘盈入库、暂存入库、盘点、旧件出库、盘亏毁损、暂存出库、旧件发票、旧件销毁清单、直接返供应商下账单和报表部分。限于篇幅，本文对报表部分暂不画出，类图只画出本模块 UI 层中主体功能类图。

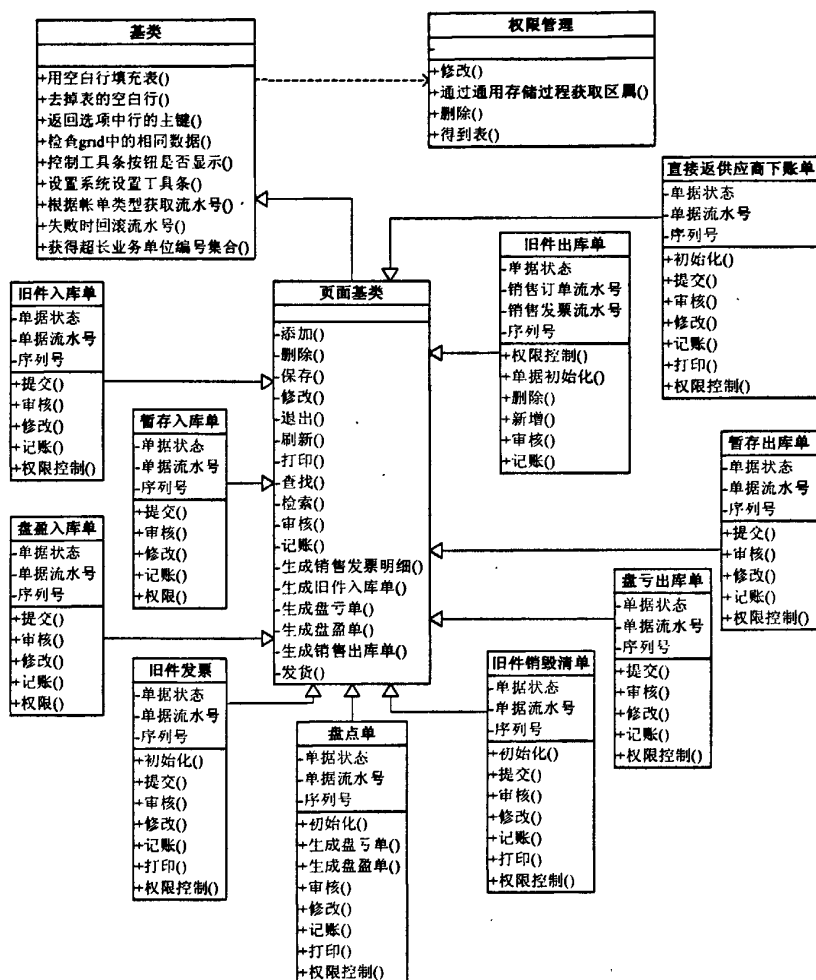


图 3-7 旧件管理类图

3.3 系统基础设置设计

基础设置管理是配件管理系统的基础模块，该模块的功能是对系统企业中单位信息的维护，通过该模块可以将配件基本信息进行综合，将各种配件的基本据添加到系统中去，基础信息是整个配件管理系统的基础，是为整个系统的配件管理做数据准备。

配件基础信息管理包括服务部仓库档案、服务部仓库库位、业务员仓库档案、往来单位档案、配件分类代码表、配件计量单位代码表、往来单位类别代码表、片区代码表、车型代码表、配件档案等信息的管理。服务部仓库档案包括配件所要存放的服务部仓库信息，是厂内中心库、旧件库还是其他

仓库。服务部仓库库位表示存放地点是几号仓库。业务员仓库档案包括指业务员仓库编号、仓库名称、所属地区、业务员姓名、联系方式的信息。往来单位档案包括往来单位编号、财务编号、单位名称、往来单位的类型、片区名称的信息。配件分类代码表包括配件代码、配件类型名称、备注的信息。配件计量单位代码表包括计量单位编号、计量单位的信息、往来单位类别代码表包括往来单位类型编号、是什么类型，如厂内中心库、区域二级库、业务员仓库、配件经销商、供应商等。片区代码表包括片区编号、片区名称、负责人基本信息。车型代码表包括编号、车型名称的信息。以上所有基本档案信息都是为配件档案提供数据支持的。而配件档案是对配件信息首先进行分类，每一类型的配件包括不同的型号的配件的新增、修改、删除、查看、导出到 EXCEL 等功能。它是以上述配件所有基础信息为基准的。其中每一类配件信息包括配件编号、配件名称、规格型号、旧编码、类型名称、计量单位、计划价、批发价、零售价、存放地点、备注等信息。

配件基础信息管理的功能结构图如下图 3-8:

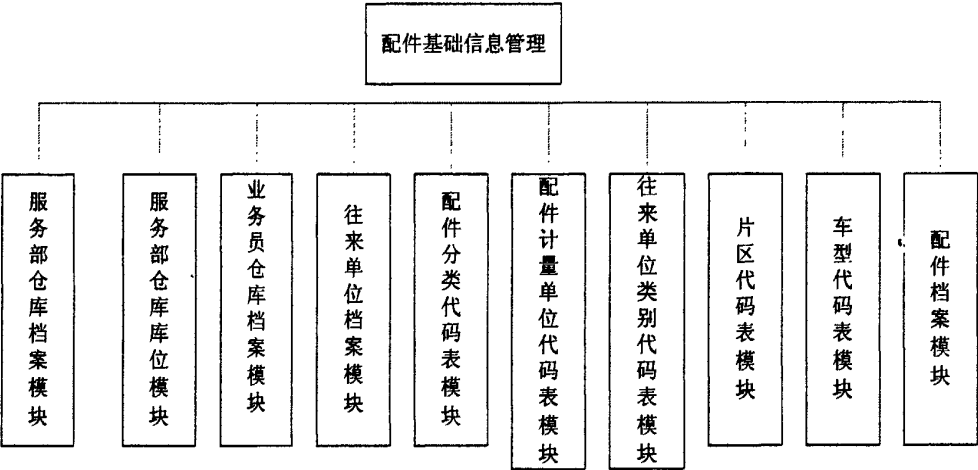


图 3-8 配件基础信息管理结构图

第四章 系统流程管理的详细设计

4.1 配件管理系统流程特征分析

4.1.1 工作流的概念与特征

工作流的概念目前还没有一个统一、明确的定义^[20]。工作流管理联盟 WFMC (Workflow Management Coalition) 将其定义为: 业务流程全部的或者部分的自动化运行, 在业务执行过程中, 文档、信息或者任务遵照一定的规则进行流转, 实现业务组织成员之间的协调工作, 以达到业务的整体目标。

流程一般具有以下几个特征^[21]:

(1) 案例: 处理案例是工作流的基本目标。订单申请、保险索赔、请假等都属于案例。相似的案例可以划为同一种案例类型。这些案例可以按照相同的方式来进行处理。

(2) 任务: 一个工作的逻辑单元就是任务, 它必须保证完整执行而不能被分割, 如果有任何错误在任务执行的期间发生, 那么任务必须恢复到被执行前的状态, 这就是“回滚”(rollback)。任务并非案例活动的一次具体执行, 一般的工作单元都可以称作任务。而下面引入“工作项”和“活动”的概念就是为了避免把“任务”与作为某案例中一部分的“任务执行”相互混淆而提出的。将要执行任务和案例的结合体就是工作项, 只要案例状态允许, 就可以创建工作项。而“活动”指的是工作项的执行动作。当工作项开始时, 它这个动作就成为一个活动。

(3) 过程: 可以通过不同的过程来对应描述每一种类型案例的执行过程。过程用来指出哪些任务需要被执行以及按照什么顺序执行。过程甚至可以被看作是对应于某种案例类型的程序。

(4) 路由: 任务以某种次序执行而产生路由, 路由的执行方式有四种: 顺序执行路由 (Sequence Routing)、并行执行路由 (Parallel Routing 两个任务同时开始于 AND-split 节点, 然后重新汇聚于 AND-join 节点)、条件路由 (Conditional Routing 任务有条件开始于 OR-split 节点, 汇合于 OR-join 节点)、自循环路由 (Self-Cycle) 一个任务自己反复执行, 直到满足条件退出)。

(5) 组织性: 一个业务流程, 一定是以完成某件工作为前提的, 一个人可以完成的工作不能称之为工作流, 所以一件流程性的工作必然是有组织性

的, 由多人完成的工作。

4.1.2 配件管理系统流程特性分析

研究配件管理系统的流程特征, 就不得不分析一下现有的几种流程^[22]:

(1) 结构化流程: 指的是实际工作过程相对固定, 严格按照某个安排好的的步骤进行, 但是业务执行过程中会反复重复某种活动的工作流。通过详细的分析此种工作流的业务过程, 可以得到定义其流程的各种类型的信息, 这些定义过的完整过程实例在以后的应用过程中可以反复的被调用。大多数办公程序, 如公文审批、处理等都属此类。

(2) 即席化流程: 指那些流程重复性不强或没有重复性的工作过程, 这种工作流执行过程中所需的有关参数(如参加者、活动等)事先不能确定, 必须推迟到过程实例运行时才可以确定, 同时在业务过程执行中还可能会发生一些突发的情况。这种动态多变的特点使得业务流程重组的建模与执行更加复杂。

(3) 以文档为主的流程: 是指利用将电子文档、图像等在业务相关人员之间进行分发的技术, 通过让不同人的对文档处理与审阅, 实现工作流程管理。目前的文档管理与映像管理系统均属此类。

(4) 以过程为主的流程: 在面向过程的工作流管理系统中, 业务流程被切割成一个个数据对象, 工作流按照每个数据对象形成的环节进行流转。系统相关人员根据分发的数据对象进行操作。例如可以将数据对象的值作为控制条件、或者将其封装成活动项等。许多专业领域的工作流管理系统均属此类。

(5) 邮件通知的流程: 是指在过程实例执行中, 消息的传递、数据的分发与事件的通知是以电子邮件来完成的。许多轻量级工作流管理系统经常使用此种方法, 它可以使整个系统将运行于一种松耦合的模式, 充分发挥电子邮件系统在广域环境下的数据分发功能。

(6) 基于数据库的流程: 基于数据库的工作流管理系统中, 所有的数据都保存在某种数据库中, 过程的执行实际上就是对这些数据的查询与处理。

(7) 任务驱动的流程: 指的是从过程的开始逐步地按照步骤的次序执行下去, 当处理完某个活动实例之后, 将激活创建后续的有关活动,

直到整个工作流程的完成。目前大多数面向过程的工作流管理系统的执行都采用这种方式。

(8) 目标拉动的流程：目标拉动的工作流管理系统中，一个业务流程被看作一个目标。过程实例执行时，该目标将被分解得到多个可执行的环节，它们相互之间按一定约束条件关联起来，各环节还可以当成是子目标进一步分解。整个业务流程执行过程就是各个环节的执行过程。

配件管理系统针对于汽车制造业有其专业性，配件管理过程必须符合规范。配件管理业务过程执行过程中，不同的案例有很多相同或者相似的活动，比如：审核、记账等业务，业务过程重复性较强。业务活动可以以接口的形式进行调用加以重新排列。在业务过程中对任务活动的处理就是对业务数据的处理，系统相关人员对业务流程的操作本质就是对系统的数据对象的操作。业务流转过程中，步骤之间消息的触发方式采用邮件进行传递。系统流程实例一旦被调用，系统的业务流程将按照步骤依次执行下去。根据以上特点结合企业业务需求，本论文研究开发的基于工作流的配件管理系统是结构化的、面向业务流程的、邮件通知的、任务驱动的轻量级工作流管理系统。

4.1.3 工作流管理系统中的人机交互

在工作流管理系统中，完成整个业务流程的主体对象是人和系统。工作流程定义工具、工作流模板都是为了完成业务过程重组和支持系统使用人员所提供的运行环境和工具。在整个流程的执行过程中，不同的操作人员需要完成的工作大致可以分为以下几种：

(1) 模型定义：建立、修改和发布企业的业务过程模型^[23]，这一般都是有系统的超级管理员或者管理员来按照企业的业务流程调整来完成。

(2) 人机交互^[24]：按照工作流模板定义所定义的工作流任务项，完成具体的业务处理工作（比如订单提交、审核、记账、查询等），这部分工作一般由企业内部进行业务操作的人员来执行。

(3) 系统运行状态的监控：用来监控和处理系统的执行^[25]。对于系统中出现的一些意外情况，比如突发性状态，需要终止、挂起、恢复某个过程实例的执行，可以对系统的执行情况进行改变。一般在企业中有较高职务的人员才有执行这个操作的权限。

对应以上分析本系统人机交互对应的逻辑关系如下图 4-1：

(1) 本系统用户分为企业决策者、超级管理员、系统管理员、部门主管、员工四种类型。

(2) 企业决策者进行业务调整, 决定业务流程管理方式并授权给超级管理员, 超级管理员和管理员一般为企事业专门负责信息化的人员, 超级管理员根据业务需要进行系统的业务调整。超级管理员负责定制、添加模板, 进行模板管理。超级管理员可以创建一个或多个模板留待调用。

(3) 系统管理员负责对业务流程进行监控和管理, 包括开启、挂起、恢复等操作。当要对流程进行重新配置时, 为防止流程冲突、异常现象, 管理员会负责关闭所有配件管理系统的客户端, 重新配置流程后, 再开启系统进行工作。系统管理员还负责对定义好的业务流程进行权限分配, 即分配每个业务步骤应该由哪种角色来负责操作。

(4) 一个用户可以担任多种角色, 一种角色下有多个用户。一种角色可以具有一个以上的功能权限, 可以将用户的权限按表单的功能进行划分。

(5) 一个流程模板包含多个流程步骤。每个实例步骤可能产生一个或多个工作项。

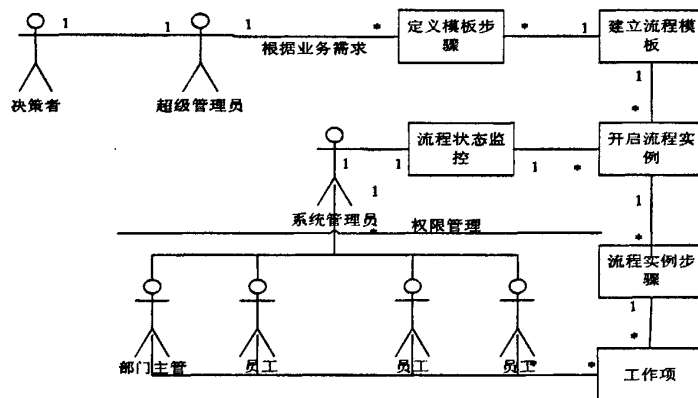


图 4-1 workflow 管理系统人机交互关系图

4.1.4 配件管理系统流程执行过程分析

下图 4-2 是系统流程执行时序图, 系统流程执行步骤如下:

(1) 超级管理员根据业务需要创建系统流程步骤, 生成模板实例, 存入模板数据库。

(2) 管理员通过 XML 获取已定义好的模板实例, 并且根据业务需要选择合适的模板实例。

(3) 模板实例生成完整的活动节点, 系统获得流程上下文签名, 并且将

信息进行初始化，存入数据库。

(4) 流程开始执行，根据流程定义找到开始节点，并且依靠任务驱动寻找下一个活动节点。活动实例的上下文由过程实例建立此活动时负责初始化。任务节点进行到 END-split 时结束。

(5) 系统内部 workflow 引擎负责分派活动给过程执行者，并且按照每次活动的结果签名返回给过程执行者，并且执行下一个节点任务。

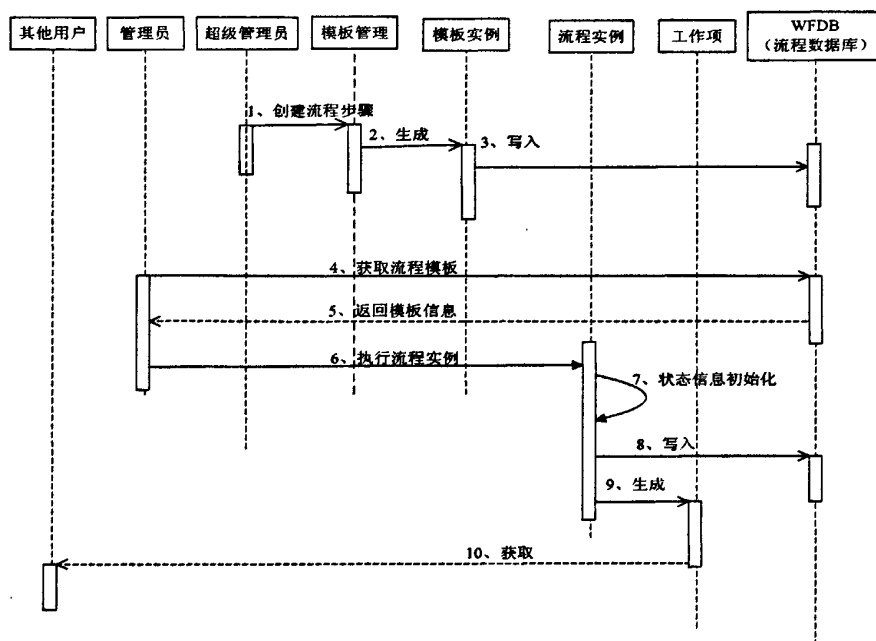


图 4-2 系统流程执行时序图

4.2 系统流程管理详细设计

本节主要介绍 workflow 管理的实现过程，系统流程管理分为 workflow 任务管理器和工作流模板定义器设计。由于本系统是轻量级基于 workflow 配件管理系统，属于为专门应用所定制的应用级软件系统，包括内置比较完整的工作流功能，这种系统是直接面向最终用户的流程化应用。所以对 workflow 管理客户端的设计，就包含了对系统的工作流引擎的设计。系统流程管理的功能包括：

(1) 接受外部对流程实例运行提交的任务，负责对过程定义进行解释，推进过程的执行，完成活动参与者与系统之间的交互，包括下达任务给用户、为用户准备任务数据和程序、接收用户提交的结果等。

(2) 根据模板生成流程实例，将活动进行处理生成用户的任务，解析步

骤跳转条件,调用各种应用,维护应用程序间或用户间传递流程的相关数据。

(3) 通过流程监控、对流程实例进行管理。实现对工作流执行状态的监控。对工作流实例进行启动、挂起/恢复、终止和删除操作,监控工作流实例的运行状态,处理工作流执行过程中的异常情况。

(4) 对系统调用的流程实例进行历史信息记录,管理日志等等。

企业业务发生重组,需要一个新的流程。超级管理员根据业务流程添加一个新的流程模板。管理员可以对新添加的模板进行查询,并且查看新添加模板的具体步骤,了解流程的具体流转情况。包括业务流程有哪些环节,有什么人来进行处理,处理的是什么业务类型。在查看过后可以根据模板开始一个流程实例。上级主管可以在流程不适合企业业务的时候终止流程运行,正处于流转过程中的业务就会中断。上级主管也可以对流程实例进行挂起操作,企业业务进入暂停状态,此时如果调用其他模板实例,新业务将按照新的流程开始。挂起的业务只有在流程实例重新恢复时才可以继续进行。

4.2.1 流程模板管理设计

流程模板管理中,超级管理员可以定义流程模板定制流程各步骤之间的关系,流程模板待管理员进行调用,开启流程定义。通过分析,流程模板与流程模板步骤之间是一对多的关系。管理员可以将系统中不同模块的业务需求定义为不同的模板实例,通过权限管理中对系统表单级权限的控制,在模板的步骤间建立起每一个步骤活动的操作者。然后开始这个流程实例。系统模块按照这个流程运行,每一个步骤结束,按照流程实例自动触发下一个步骤,下一步任务的操作者可以对这个处理完的业务进行处理。

图 4-3 是流程模板管理功能结构图。由上面分析可以得知流程模板管理包括流程模板步骤定义和流程步骤约束两个方面,流程模板步骤定义依赖流程步骤约束,即每一个步骤都需要一个用户角色来执行。

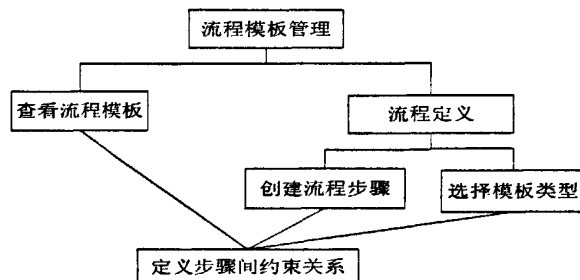


图 4-3 流程模板管理功能结构图

选择要建立系统模块的模板后，系统根据模板对应模板 id 提取出相应一系列步骤节点，由用户根据业务需求进行节点选择。在定义步骤的同时应该选择好每一步骤的负责人角色，每一个角色有哪些负责人。所以，模板数据项应该包含模板编号、模块类型 id、步骤名称、步骤序号、步骤操作人等基本信息。

根据企业业务需求，流程模板步骤约束间可以通过用户角色进行约束。这样可以清晰的划分出流程步骤之间操作角色的关系。流程步骤将严格按照步骤操作角色执行下去。通过对其功能进行抽象，可以画出如下图 4-4 流程管理模板类图。类图中一种模板类型将包含多个流程模板，一个流程模板包含多个模板步骤，一个用户角色包含多个用户，所以这些类之间的都是组合关系。其他的类之间如：创建流程模板时要获得用户角色权限，需要调用数据库访问类，所以这些类之间都是类的依赖关系。

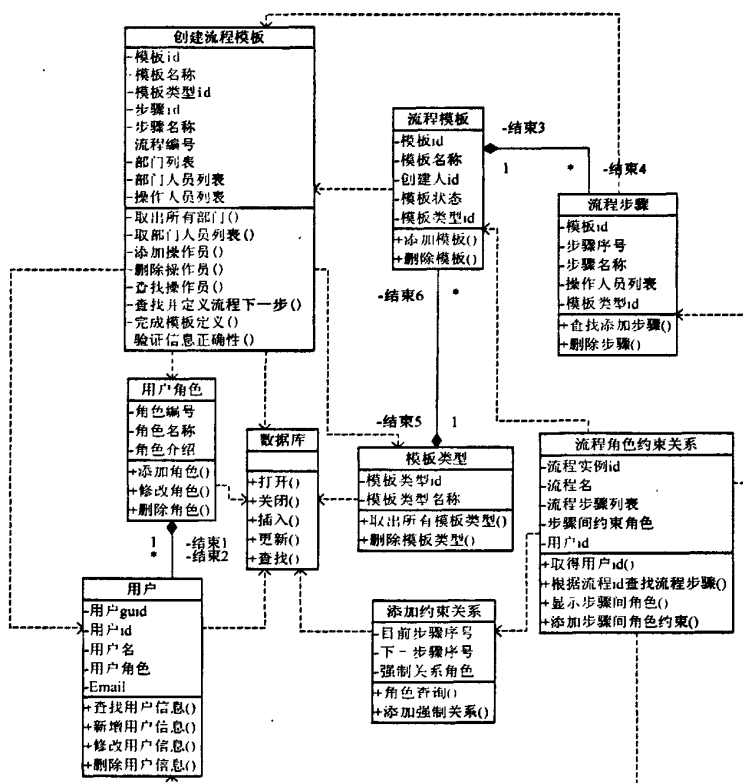


图 4-4 流程模板管理类图

4.2.2 流程任务管理设计

本系统采用 C/S 结构，需要访问远程数据库，为了避免系统在流程定义

过程中因数据库连接出现的错误以及提高系统的反应速度,本系统在流程步骤的跳转上采用满足 XPD L 规范 XML 的方式,将流程步骤存储在 XML 文档之中。管理员对流程模板进行调用,当调用某个流程模板运行流程实例的时候,系统会在 SERVER 端编译生成 XML 文档,每一次的流程实例调用,都会更新 XML 文件。通过将流程步骤放在 XML 文档中的机制,可以有效防止系统因意外死机或数据库断开而导致的流程调用异常,同时可以降低连接数据库的次数。而数据库中流程实例表中的流程数据只起历史流程数据备份的作用,为历史业务数据的查询,管理的决策起指导性作用。如图 4-5 流程实例定义过程所示,操作员操作流程任务管理器,调用流程模板运行流程实例,将流程实例步骤信息进行编译存入 XML 文档,然后将实例信息保存在数据库中,而 XML 文档中的流程实例信息是对数据库中流程模板的信息进行加载的,然后将业务流程步骤展现给用户。

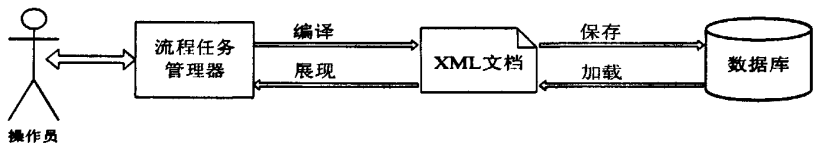


图 4-5 流程实例定义过程示意图

流程实例是根据流程模板来生成的托管系统流程状态的工具,一个流程实例的运行,则意味着系统正在按照这个实例的流程运行,由于流程模板定义的实例非常多,所以需要有一个工具来管理这些数据,这就是流程任务管理。流程实例有两个作用,一个是查看,一个是控制。查看是用来分析流程实例的运行情况,包括流程步骤的执行者的信息等。控制则是对流程实例的状态进行控制,如对流程的终止、挂起/恢复、开始等。流程实例删除是对流程模板中实例的删除。

由此可以得出如图 4-6 系统流程任务管理功能结构图:

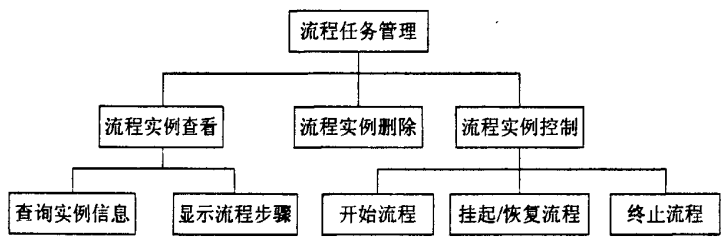


图 4-6 流程任务管理功能结构图

“流程实例查看”根据业务类型和流程实例状态来查询存在于系统中的流程实例信息以及流程步骤。流程实例控制中“开始流程”用于系统业务重

组，开启系统中一个业务模块的新流程。“挂起流程实例”子模块用于暂停流程的运行，通过“恢复流程”可以讲流程回复到运行状态。“终止流程实例”用于彻底终止系统现行的业务流程，将流程处于结束状态，不可恢复。

流程定义就是制定模板指定的系统业务流转规则流程。这些规则就是根据流程实例中的数据所制定的流转条件。所谓的流程自动处理，其实就是对这些条件加以解释，按照指定的业务流程进行流转。控制流程实例、活动、工作项的状态转换，处理业务数据在各个流程步骤之间的传递和数据转换。

用户根据创建好的流程模板进行流程实例定义。当用户选择好流程模板, 开启一个业务流程后, 系统将流程信息存入数据库, 包括流程开启人、流程编号、流程类型、模板编号、开始时间等。然后根据此流程实例提取出该流程所有的步骤, 把步骤存入数据库表, 包括流程编号、步骤名称、步骤序号等信息, 同时更新 XML 文档。系统用户进入系统, 系统取出该用户信息, 包括用户 Id, 用户名, Email 等信息放入哈希表。系统根据 XML 中流程信息进行流程判断, 判断当前业务流程流向, 如先提交, 再审核, 再记账。其他用户登录时提取出用户权限将系统菜单功能初始化, 用户只能看到自己权限所能看到的菜单项。用户对单据进行操作, 在对系统进行业务操作时可以选择是否用邮件提醒下一个操作者优先处理单据, 如果选择邮件提醒, 系统将自动发送 Email 给下一步操作者提醒他进行下一步操作, 用户操作完成后单据项关闭。通过以上的设计, 可以得出图 4-7 流程任务管理类图:

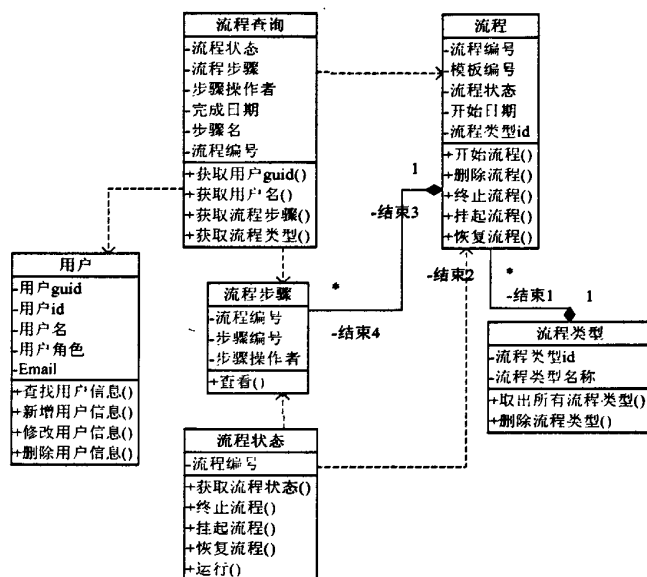


图 4-7 流程任务管理类图

4.3 数据库设计

4.3.1 数据库概念结构设计

数据的概念模型设计是数据设计中的重要步骤之一^[26]。它将现实中存在的客观对象抽象成不依赖于任何具体机器或者平台的信息结构,它并不是计算机所支持的数据模型,而只是存在于概念上的模型,然后利用这个模型,转化为计算机所支持的数据模型,并且这种概念模型,是很容易被转化为关系、网状、层次等数据模型的^{[27][28]}。

数据模型的几个要素的具体概念如下^[29]:

(1) 实体:任何可以相互之间区别并且客观存在的对象都可以成为实体,他们不仅仅是具体的事物,也可以是抽象的概念或联系。例如:订单、用户权限、流程模板等。

(2) 属性:实体所具有的某一特性或特征称为属性。每一个实体都可以用一个或一个以上的属性来描述。

(3) 联系:实体内部以及实体之间都是有联系的。实体内部的联系指组成实体的属性之间的联系;实体之间的联系是指不同的实体之间的联系。实体之间的联系可以分为三类:一对一联系(1:1);一对多联系(1:n);多对多联系(m:n)。

根据前面对系统的需求分析以及前面对流程管理客户端功能模块的设计,可以抽取出来的实体有:流程类型实体、流程模板实体、日志实体、用户实体、用户权限实体、用户角色实体、模板步骤实体、流程实体、流程步骤实体。他们的关系可以总结如下:模板步骤和模板约束是多对多的关系,一个流程类型可以有多个流程模板,一个用户可以创建多个流程模板。流程模板对应一个流程实例,一个流程包含多个步骤。一个用户可以开启多个流程。一个角色可以包含多个用户。一个角色可以有多种不同的权限,一个用户可以操作多个订单,一个订单的完成需要多个用户来操作,一个订单的单头对应一个订单的详细表。

下图是销售订单与流程管理模块的数据模型设计,其他系统模块的数据概念模型以此为例,就可以得到系统全部的数据库模型。

流程管理模块 E-R 图如下图 4-8:

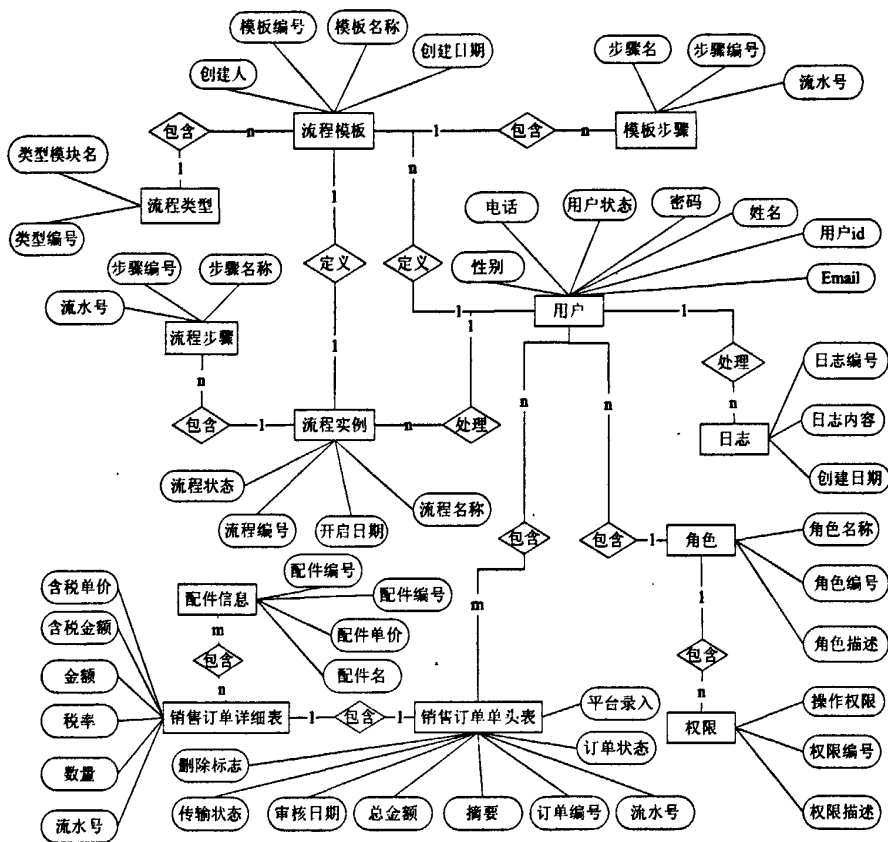


图 4-8 流程管理 E-R 图

4.3.2 数据库逻辑结构设计

在系统数据库设计过程中，数据表的设计始终以业务为核心。为此，可以抽取出与业务联系最紧密的几张表作为主表，与主表相对应的是副表，最后，用关系表将他们相关联起来。由于本系统研究的是基于工作流的汽车配件管理系统，且销售模块涉及到的业务较复杂比较多，限于篇幅下面以配件订单模块中的销售订单涉及到的部分表和流程管理客户端的部分表为例，抽取出一部分数据库表如下表。

表4-1 销售订单单头表

名称	字段名	数据类型	预设	非空	主键	外键
订单流水号	BillGuid	varchar(36)		1	1	
客户编号	Customer Id	varchar(36)		1		1
订单编号	BillId	varchar(50)		1		
订单录入日期	BillInputDate	datetime		1		

业务员编号	BusinessmenId	varchar(36)		1		1
客户姓名	Customer	varchar(20)				
客户电话	CustTeleNum	varchar(20)				
交货地点	TradeLocation	varchar(100)				
备注	Remark	varchar(200)				
总金额	SumMoney	money	0	1		
主管编号	DirectorId	varchar(36)				
主管	DirectorName	varchar(20)				
制单人编号	OperatorId	varchar(36)		1		1
制单人姓名	OperatorName	varchar(20)				
制单时间	OperatorTime	datetime		1		
审核人编号	AssessorId	varchar(36)				
审核人姓名	Assessor	varchar(20)				
审核时间	AuditDate	datetime				
是否平台录入	FlatInputState	bit				
删除标志	DelState	bit	0			
数据传输状态	TransState	varchar(20)	未上传			

表 4-2 订单明细表

名称	字段名	数据类型	预设	非空	主键	外键
流水号	BillDetailId	varchar(36)		1	1	
业务流水号	BillGuid	varchar(36)				1
配件 ID	AssemId	varchar(36)				1
数量	Amount	float	0	1		
单价	UnitPrice	money				
金额	SumMoney	money				
税率	Cess	numeric(5,2)		1		
含税单价	TaxUnitPrice	money	0	1		
含税金额	TaxMoney	money	0	1		
数据传输状态	TransState	varchar(20)	未上传			

订单单头表和订单明细表是两个主要业务表，而在订单制定流程中需要查询配件经销商的余额帐，这样可以找出它们之间的联系，下面是经销商余额帐表。

表 4-3 经销商余额账表

名称	字段名	数据类型	预设	非空	主键	外键
PKID	BillGuid	varchar(36)	newid()	1	1	
经销商编号	FranchiserId	varchar(36)				1
期初	Account	money				

本期收入	CurrIncome	money		1		
本期支出	CurrPayOut	money		1		
结余	Balance	money				
备注	Remark	varchar(200)				
删除标志	DelState	bit	0	1		
数据传输状态	TransState	varchar(20)	未上传			

与以上数据库表的设计类似，流程管理模块表的设计也是这样，首先设计出流程管理模块的主表，然后设计出副表和关系表,这里给出流程管理的部分数据库表。

表 4-4 流程模板定义表

名称	字段名	数据类型	预设	强制	主键	外键
模板编号	FlowModelId	varchar(36)	newid()	1	1	
模板名称	FlowModelName	varchar(36)				
添加时间	CreatTime	Datetime(8)				
用户编号	UserGuid	varchar(36)		1		1
模板类型 id	TypeId	varchar(36)		1		1

表 4-5 模板步骤定义表

名称	代码	数据类型	预设	强制	主键	外键
模板编号	FlowModelId	varchar(36)		1		1
步骤序号	StepId	Int (4)				
步骤名称	StepName	varchar(255)				
角色 id	RoleId	varchar(36)		1		1
步骤关系 id	StepRelationId	varchar(36)		1		1

表 4-6 流程定义表

名称	代码	数据类型	预设	强制	主键	外键
PKID	Pkid	varchar(36)	newid()	1	1	
开启人 id	UserId	varchar(36)		1		1
流程 id	ModelId	varchar(36)		1		1
流程状态	FlowState	Int(4)		1		
开启日期	RunTime	Datetime(8)				

表 4-7 流程步骤表

名称	代码	数据类型	预设	强制	主键	外键
PKID	PKID	varchar(36)	newid()	1	1	
流程 id	ModelId	varchar(36)		1		1
步骤编号	StepId	varchar(36)		1		
角色编号	RoleId	varchar(36)		1		1
步骤流水号	FlowSeq	Int(4)		1		

本系统的三个业务模块的单据繁多，业务复杂，但是表的结构有很多相

似, 本文依据数据库原理中的第三范式原则, 数据库表 PKID 指各个表的主键, 用来区分不同的记录, 表和表通过外键相关联, 确定表之间的逻辑关系。限于篇幅, 这里只给出了用户流程管理中的一部分数据表逻辑关系图。如图 4-9 流程管理数据库表逻辑关系图:

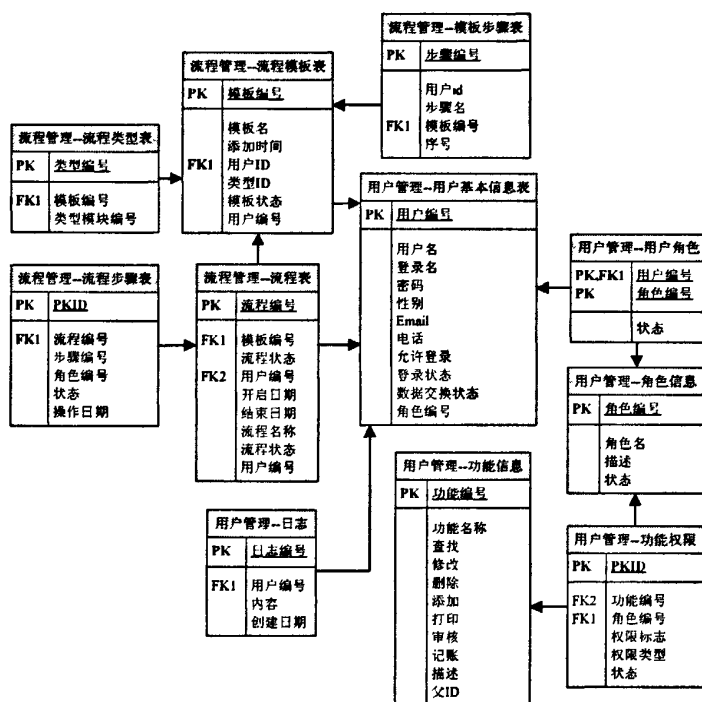


图 4-9 流程管理数据库表逻辑关系图

4.3.3 数据库存储过程的使用

存储过程(Store Procedure)是一组用于执行数据库中特定功能的 SQL 语句集, 它以 SQL 语句预编译的形式存储在数据库中。用户使用它的时候只需通过调用存储过程名并给出参数(如果存储过程本身有参数)来执行^[30]。

存储过程与一般的 SQL 语句相比较, 具有很多的优点。当一个存储过程首次运行时, SQL Server 查询分析器将会对它进行预编译, 并且为它建立优化的查询方案。以后对该存储过程的调用中, 不需要再次对它进行优化, 而是直接调用高速缓存中的优化查询方案。这样就提高了它的执行速度。相对于一般的 SQL 语句和批处理语句, 存储过程有以下优点^[31]。

(1) 存储过程准许标准组件式编程。存储过程经过编译, 在程序中不用重新编写该存储过程的 SQL 语句, 创建后的存储过程可以被多次调用, 达

到重用的目的。数据库专业人员可以随时对存储过程进行修改,但对调用存储过程的程序源代码没有影响,程序源代码只是起调用的作用和业务逻辑作用,从而极大地提高了程序的可移植性。

(2) 由于存储过程是预编译的 SQL 语句,同时 SQL Server 查询优化器对其进行分析优化,最终给出其在系统表中的执行计划。所以在运行过程中,存储过程可以以较快的速度执行。

(3) 存储过程在被调用的时候,数据库会把预编译的 SQL 语句在服务器端进行处理,并且把处理的结果返回给客户端,这样做有利于客户端和服务器的负载均衡,否则如果使用大量的 SQL 语句执行,将会大大的减少网络流量,降低网络的负载和系统对数据库的访问速度,存储过程代码直接存储于数据库中,就不会产生大量 T-sql 语句的代码流量。尤其在多用户同时访问时,效率提升非常突出。

(4) 存储过程还可以作为一种安全机制来加以利用,系统管理员通过对执行的存储过程进行权限的限制,就可以实现用户对其相应的数据访问的限制,这样也保障了数据库数据的安全,避免了非法用户对数据的访问^[22]。

本系统在开发过程中使用了大量的存储过程,下面简单举例说明本系统使用的存储过程。存储过程实例:根据用户 ID 和角色名称查找角色下的权限集合。

```
CREATE PROCEDURE [dbo].[Proc_wfGetEmpRight]
@empID VARCHAR(36), @funcCode varchar(50)
AS
SELECT R.RightType,MAX(rightflag) AS RightFlag
FROM wfFuncRights R
INNER JOIN wfEmpRoles E ON R.RoleID = E.RoleID
INNER JOIN wfFuncDic F ON F.FuncID = R.FuncID
WHERE (E.EmpID = @ empID)
AND FuncCode = @ funcCode AND (R.DelState = 0) AND (E. DelState = 0)
GROUP BY R.RightType
```

第五章 基于工作流的汽车配件管理系统的实现

5.1 开发环境与工具

(1) 操作系统

客户端操作系统采用 Windows XP/NT/2000 系列操作系统, 服务器操作系统采用 Windows2003 server。

(2) 系统数据库平台

数据库系统采用关系数据库 SQLServer2000 是由 Microsoft 公司开发的、支持对象关系模型的大型数据库产品。SQL Server2000 是一个功能完备的数据库管理系统。它包括支持开发的引擎、标准的 SQL 语言、扩展的特性(如复制、OLAP、分析)等功能^[33]。

(3) 开发工具

系统采用的开发工具 Microsoft Visual Studio.Net2005^[34](VS2005)是 VS2003 的升级产品, VS2005 作为一个功能强大的可视化集成开发环境支持多种语言的开发, 例如 C#、VB、Java 等^[35]。它在改善操作的同时极大的提高了开发人员的效率, 它提供的类库资源非常多, 相对 VS2003 做了非常多的改变, 其中最重要的是将原运行类库(.NET Framework)从 1.1 升级到了 2.0^[36]。

(4) UML 设计工具

采用 RationalRose2003^[37]为系统建立 UML 模型, 设计了系统用例图和系统的部分类图^[38]。

(5) 数据库设计工具

系统数据库利用 Sybase Power Designer 作为辅助设计工具^{[39][40]}, 利用它可以制作数据流程图、概念数据模型、物理数据模型, 可以生成多种客户端开发工具的应用程序, 还可为数据仓库制作结构模型, 也能对团队设计模型进行控制^[41], 本系统数据库完全由 Power Designer 设计以及生成。

5.2 系统部分功能的实现

限于篇幅, 在实现部分, 论文仅对系统的部分功能的实现加以详细描述。

5.2.1 系统数据库访问的实现

本系统采用了微软提供的 SqlHelper 类进行数据库访问。SqlHelper 类提供了一组封装好的静态方法，系统通过它向 SQL Server 数据库发出许多各种不同类型的命令^[42]。其中 SqlHelperParameterCache 类用来将命令参数放入缓存，利用这种方式提高性能。该类在内部使用了很多的 Execute 方法（尤其是那些运行在存储过程中重写了的方法）。当数据访问客户端时，可以直接使用它缓存特定命令的特定参数集^[43]。SqlHelper 使用 SqlHelperParameterCache 类来对参数进行管理，它提供了三种方式的公共共享方法来管理参数。它们是：CacheParameterSet：它用于将 SqlParameter 数组存入缓存中；GetCachedParameterSet：它负责查询存储在缓存中的参数数组副本；GetSpParameterSet：它作为一种重载方法用来检索指定存储过程中的参数，它的检索方式是先查询一次数据库，然后将查询结果放在缓存中，增加以后的查询效率。对于 C# 语言，SqlHelper 类提供了五种最基本的共享 static 方法，它们是：ExecuteNonQuery、ExecuteDataset、ExecuteReader、ExecuteScalar 和 ExecuteXmlReader。他们用来实现的每种方法都提供了一组相同的重载。这种方式提供了一种很好的模式，它主要依靠使用 SqlHelper 类来执行命令，为开发人员在数据库的数据访问方式上提供了必要的灵活性。这些方法的重载都支持不同的方法参数，这样开发人员就可以根据方法确定传递连接、参数信息、事务的方式。本系统利用了 SqlHelper 类编写了基础的数据访问类和方法，并以此作为数据库访问的基础。这样使系统业务逻辑层只专注于对系统业务逻辑的分析与设计，只在需要数据时与数据访问层联系，这样实现了系统的解耦。

SqlHelper 作为一个封装好的访问数据库的类，其本质就是利用 ADO.NET 访问数据库，所以在使用 SqlHelper 的时候需要首先加入 SqlHelper 类，然后在项目中添加引用 System.Data，在数据库操作的基类中需要使用 using 添加 System.Data; System.Data.SqlClient 两个命名空间。

以下把 hash 表中的数据，保存到数据库为例，给出主要代码实现，其中使用 SqlHelper 中的 ExecuteNonQuery 方法调用了存储过程。

```
public class DataConn //建立数据库连接
```

```
{
```

```
    public SqlConnection getcon()
```

```
    {
```

```
        SqlConnection myCon = new SqlConnection(Str_Conn);
```

```
        return myCon;
    }
    public string Str_Conn
    { get
      { Return
System.Configuration.ConfigurationManager.AppSettings["ConnectionString"];
      }//读取系统app.config中字符串连接信息
    }
}
/// 把hash表中的数据，保存到数据库
public bool ModifyDataTable(Hashtable hashTable, string spName) //spName为
存储过程名，hashTable为哈希表
{ try
{
SqlParameter[] myParams =
SqlHelper.ParameterCache.GetSpParameterSet(sqlConn, spName);
ParamAssignValue(ref myParams, hashTable);// sqlConn为实例化DataConn
的对象
int count = SqlHelper.ExecuteNonQuery(sqlConn,
CommandType.StoredProcedure, spName, myParams);
    if (count > 0)
        return true; /// 返回影响行数
    else
        return false;
}
catch (Exception ex)
{
    throw new Exception(ex.Message);
}
finally
{
    CloseSqlConnect();
}
}
```

5.2.2 系统界面整体效果的实现

为了系统的美观以及系统功能更好的搭建，本系统的所有界面均继承自

ToolBarForm 类, 而 ToolBarForm 类继承自页面基类 baseForm 类, 页面基类继承了 System.Windows.Forms.Form, 这样就做到了界面显示风格的统一。为了实现用户对表单级别的控制, 系统编写了用户控件 ToolBar 类, 通过 baseForm 类对 ToolBar 类的调用达到了用户各级表单对单头功能的统一权限控制。ToolBar 类中为表头的各个功能定义了接口, 在 baseForm 中添加了很多对表进行设计的基本方法。而为了实现系统的表单级的功能权限控制, 又编写了工具条类 ToolBarForm, 这个类继承了 baseForm, 并且在类中定义了很多的抽象事件接口, 在主菜单和其他业务菜单中只需要继承 ToolBarForm 类, 并且对 ToolBarForm 中的方法进行重写, 这样做的好处是用户权限控制直接操纵 ToolBarForm 类中的功能控件, 在用户角色对功能的分配上可以通过控制 ToolBarForm 中的功能按钮, 就可以做到用户对不同的表单的不同功能加以控制。另外如果用户想添加其他的业务功能, 只需要在 ToolBarForm 写入空方法, 其他类在继承的时候进行方法的重写即可, 体现了面向抽象编程的思想。而且由于多有的单据继承了 ToolBarForm, 也统一了 Form 的颜色和所有功能按钮的风格, 并且经过反复的调用, 做到了代码的重用, 统一了系统的风格。通过对表单的功能控制来实现对用户的权限控制的函数方法代码见附录。

5.2.3 流程实例编译的实现

前面设计部分提到, 用户进行流程模板的定义, 当用户完成流程模板的定义与对流程步骤的角色分配, 并且运行流程实例的时候, 系统对流程实例进行编译, 并以 XPDL(XML Process Description Language)的规范将流程步骤存放在本地 XML 文档中。当用户调用流程模板, 运行流程实例的时候需要对 XML 中的各个元素和活动节点进行解析。对 XML 进行解析的方法主要有两种: SAX(Simple API for XML)和 DOM(Document Object Model)。DOM 是根据 W3C(World Wide Web Consortium:万维网络联盟)支持的创建和处理 XML 文档结构和内容的 API。根据 XML 的层次结构, 分析 XML 文件的时候, 它会将每个 XML 结构, 如: 元素、文本、注释等都映射为一个对象, 使其在内存中映射为一棵树。SAX 是由一组精心设计的“回调函数方法”组成的用于分析 XML 的事件驱动的编程接口^[45]。它可以被看作一个底层的分析方法, 与 DOM 相比, SAX 速度较快, 内存占用较少, 但是需要较多的编程工作^{[45][46][47]}。XmlDocument 类是 XML 文档在 .NET 中的表示形式, 它

代表了 XML 文档在内存中树型结构的文档节点，即所有的节点都在文档节点下，XmlDocument 类包含一系列的 CreateXXX()方法，这些方法允许创建所有派生自 XmlNode 的类型的节点，通常将该类与 XmlNode 类一起使用就可以完成对文档的各种基本操作，该类中利用一个 Load()方法来加载 XML 文档，Load()方法的一个重载版本允许从 XmlTextReader 加载文档，这种做法带来的好处是，如果 XML 文档较大，我们可以先使用 XmlTextReader 过滤掉不相关的文档部分，这样即解决了 DOM 所带来的资源损耗问题又可以保留 DOM 对文档操控的便利性，该类中的 Save()方法用于保存文档^[48]。通过分析，本系统的 XML 中需要解析的流程活动数据不是很多，对客户端来说，其占用内存和速度的影响不大。所以采用了 DOM 的方法对 XML 文件完成解析工作。系统解析 XML 的部分函数方法代码见附录。生成的具体的 XML 文档片段如下：

<Workflow>

<WFProcess Id="20090409035828" Name="配件销售模板一"> (1)

<ProcHeader>

<Created>2009-4-09</Created> (2)

<ModelType>配件销售</ ModelType > (3)

</ProcHeader>

下面对 XML 文档中标注的部分进行说明：

(1) 系统自动根据时间生成模板序列号，模板名为“配件销售模板一”。

(2) 创建时间为 2009 年 4 月 9 日。

(3) 模板类型为配件销售模块。

<Activity Id="20090409035828_ACT_1" Name="销售订单提交"> (1)

<ActPerformer Performer="AND"> (2)

<Role Id="00077496 " Type="订单录入人员"> (3)

</Role>

</ActPerformer>

</Activity>

下面对 XML 文档中标注的部分进行说明：

(1) 定义活动的 ID 是 20090409035828_ACT_1，活动名为“销售订单提交”。

(2) 活动路由为串联(AND)。

(3) 定义本活动角色参与者的 Id 流水号和角色类型。

<Activity Id="WFM20090409035828_ACT_2" Name="销售订单审核">

```
<ActPerformer Performer="AND">
<Role Id="000774967" Type= "订单审核人员">
</Role>
</ActPerformer>
</Activity>
<Activity Id="20090409035828_ACT_3" Name="发货">
<ActPerformer Performer="AND">
<RoleId="00077500 " Type= "仓库管理人员">
</Role>
</ActPerformer>
</Activity>
<Transitions>
<Transition Id="20090409035828_TRAN_1" Name="连接"
From="20090409035828_ACT_1" To="20090409035828_ACT_2"> (1)
</Transition>
<Transition Id="20090409035828_TRAN_2" Name="连接"
From="20090409035828_ACT_2" To="20090409035828_ACT_3"> (2)
</Transition>
</Transitions>
```

下面对 XML 文档中标注的部分进行说明：

- (1) 流程步骤从"20090409035828_ACT_1"到"20090409035828_ACT_2"。
- (2) 流程步骤从"20090409035828_ACT_2"到"20090409035828_ACT_3"。

流程模板定义完毕，流程之间的步骤操作者可以由权限管理来定义，通过对系统表单级功能权限的定制，实现系统的每个步骤的操作。

5.2.4 流程步骤之间消息传递的实现

系统流程在运行的时候，在用户完成一个步骤以后需要另外一个用户进行下一步的操作。作为大型集团的业务，为保证工作的顺利进行，并且保证工作的权责分明和业务的及时处理，这里采用了邮件消息传递的方式。在本用户执行完毕的时候，如果该订单需要尽快处理，需要一个提示去告诉下一个用户尽快处理这个订单的操作，这个“提示”就由邮件来完成。当用户操作完这个步骤，如果需要下一个用户尽快处理，那么可以选择邮件通知选项，

那么系统会自动触发邮件类，去调用下一个操作者的邮件地址，给他发送一封邮件，提示他去处理下一步的业务。这种邮件的方式不仅可以及时提醒相关人员及时的处理业务，并且邮件具有不可抵赖性，更易追究相关者的责任。邮件体中嵌入用户的姓名、id，要处理的订单编号等内容，使业务流程可以更加顺利的进行，全责划分也更加分明。这里.NET 平台提供了.Net.Mail 组件，所以在邮件发送类中需要用 using 导入 System.Net.Mail 包，通过编写以下代码可以完成流程之间邮件的发送。在代码的头部，本文用注释定义了邮件体中变量的意义。当方法返回 true 时，表示邮件发送成功。邮件发送函数体如下：

/// 发送邮件

```
public bool EmailSend(string userFrom, string userTo, string mailSubject, string
mailBody, string smtpHost, string userName, string password)
{ string[] address = userTo.Split(';'); //以';'分割邮件地址,并放入数组中
bool isOk= true;
foreach (string m in address) //遍历邮件地址
{
try
{ MailMessage mailText = new MailMessage();
mailText.From = new MailAddress(userFrom); //发送人邮件地址
mailText.To.Add(new MailAddress(m.Trim())); //接收人邮件地址
mailText.Subject = mailSubject; //邮件主题
mailText.Body = mailBody; //邮件体，一般放置订单编号，操作者姓名等信息
SmtpClient scString = new SmtpClient(); //SMTP 服务器地址
scString.Host = smtpHost;
scString.UseDefaultCredentials = true; //用户匿名发送
scString.Credentials = new System.Net.NetworkCredential(userName, password);
//用户名、密码
scString.DeliveryMethod = SmtpDeliveryMethod.Network; // SMTP 传递方法
的字符串
scString.Send(mailText); //发送
}
catch
{ isOk = false; } //不成功
```

```
}  
return isOk; }//成功
```

5.2.5 流程步骤中权限分配的实现

流程的执行过程中，每一步都需要一个以上的角色进行操作，这种操作的分配就是角色的权限控制行为，通过对流程中活动功能的分配，做到用户角色对流程步骤的处理。同时，用户的权限管理可以对用户的信息进行添加删除和管理，也可以监控用户的状态。通过系统的用户管理表的增删改功能，可以对该系统的员工进行管理。通过直接在界面上对员工的用户名和登录名进行修改，点保存按钮可以修改员工的信息。系统的角色管理表中，管理员可以定义系统的角色，并且对角色加以描述。同时也可以直接在表中直接对角色进行修改，点击保存即可保存所有信息。限于篇幅，程序中调用保存事件这里不做描述，下面给出对于修改角色功能权限的存储过程。

```
CREATE PROCEDURE [dbo].[Proc_sysFuncRights_Modify]  
@FuncID int ,--功能id  
@RoleGUID VARCHAR(36) --角色id,  
@RightFlag int ,--权限开启与否的标志  
@RightType varchar(20) --功能权限类型，如“销售订单提交”等  
AS  
    IF EXISTS(SELECT * FROM sysFuncRights WHERE  
RoleGUID=@RoleGUID AND FuncID=@FuncID AND  
RightType=@RightType)--判断假如存在就更新权限状态  
    BEGIN  
        UPDATE sysFuncRights  
        SET [RightFlag] = @RightFlag  
        WHERE RoleGUID=@RoleGUID AND FuncID=@FuncID AND  
RightType=@RightType  
    END  
    ELSE—否则插入新的角色权限对应关系  
    BEGIN  
        INSERT INTO sysFuncRights  
        ([FuncID],[RoleGUID],[RightFlag],[RightType],[DelFlag])  
        VALUES  
        (@FuncID,@RoleGUID,@RightFlag,@RightType,0)  
    END  
END
```

5.2.6 设计模式在业务流程活动分发中运用

作者在对系统进行设计的时候,运用了设计模式的思想来进行系统的设计,下面举例说明抽象工厂模式在系统业务活动分发实现中的运用。使用抽象工厂模式的条件是:同属于一个产品族的产品是在一起使用的,系统提供一个产品类的库,所有的产品都以相同的接口等待调用,通过这样,使客户端不依赖于实现^{[49][50]}。由于在流程管理的活动分发中我们关注于接口的调用,并且将活动接口按产品分类,利用抽象工厂模式可以提高代码的复用度和系统的解耦,所以使用了这种设计模式。限于篇幅,这里只给出部分方法体的实现,目的在于介绍怎样将抽象工厂模式的思想运用其中来解决的问题。读者可以由下列给出的代码看出抽象工厂模式的应用。

业务逻辑判断类中的业务活动分发采用的抽象工厂模式关键代码片段如下:

```
public interface BusinessDispense //定义基础业务分发接口
{
}

public class saleSubmit: BusinessDispense
{ //销售订单提交实现基础业务分发接口
    public void saleSubmit()
    {
        ..... //“销售订单提交”业务逻辑方法
    }
    .....
}

public class invoiceSubmit: BusinessDispense //销售发票实现基础业务分发接口
{
    public void invoiceSubmit()
    {
        .....//“销售发票提交”业务逻辑方法
    }
    .....
}

public class SaleBill //配件销售业务工厂类
{
    public saleSubmit CreatSubmitAction() //创建“提交”活动
    {
```

```
        return new saleSubmit();
    }
    .....//其他业务方法
}
public class InvoiceBll//配件销售业务工厂类
{
    public invoiceSubmit CreatSubmitAction()    //创建“提交”活动
    {
        return new invoiceSubmit();
    }
    .....//其他业务方法
}
public void BusinessCall(string BusinessName)//业务调用方法
{
    BusinessDispense MyBusiness;//定义接口
    SaleBLL MySaleFactory = new SaleBLL();//实例化销售订单业务类
    InvoiceBll MyInvoiceFactory =new invoiceBll();//实例化销售发票业务类
    switch (BusinessName)//参数判断
    {
        case "销售订单提交":
            MyBusiness= MySaleFactory.CreatSubmitAction();//接口调用
            break;
        case "销售发票提交":
            MyBusiness= MyInvoiceFactory.CreatSubmitAction();
            break;
        .....//其他参数判断
        default:
            break;
    }
}
```

以上的代码里，定义了一个接口BusinessDispense，它是业务分发接口。saleSubmit和invoiceSubmit两个业务活动类实现了这个接口，SaleBll担负的作用是销售订单业务工厂类，里面利用CreatSubmitAction创建业务活动。同样InvoiceBll是销售发票的业务工厂类。当进行业务调用的时候，用户调用活动，进行逻辑判断，将BusinessName传入业务调用方法BusinessCall进行判断，通过业务分发接口MyBusiness调用各个业务活动

5.2.7 系统界面部分功能及实现效果图

(1) 流程任务管理

用户通过点击相应流程模板可以查询流程模板的详细步骤和模板状态。同时可以对流程模板进行开启、挂起、恢复、终止、删除等操作。界面效果图如下图 5-1:



图 5-1 流程任务管理界面效果图

(2) 流程模板定义

超级管理员定义流程模板，其定义的步骤在前面设计部分已经给出，这里不再赘述。需要说明的是，流程类型一旦给定，系统相应模块的流程步骤也相应给定，管理员在定义模板的时候是对这些业务进行重组，所以其定义步骤也就是对业务活动重新安排步骤的次序，进行业务重组。如：在配件销售过程中，可以先提交销售订单，再开发票，再开出库单；也可以先开销售订单，再开出库单，在开发票；也可以先开销售订单，直接开出库单结束。等等业务流程需要用户自己去定制。已经定义好的次序在步骤列表中给出。需要注意的是：用户进行流程定制必须符合业务逻辑，不可以违制操作，如：不开销售订单而直接开发货单等等不符合业务逻辑的流程定制行为。流程模板定义界面实现效果图如下图 5-2:

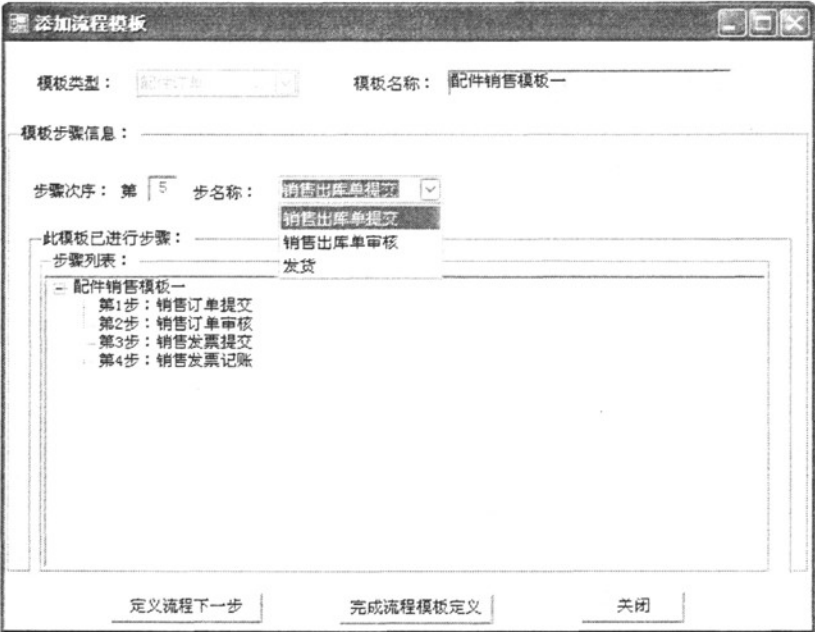


图 5-2 流程模板定义界面效果图

(3) 流程步骤权限分配

管理员可以根据分配用户角色, 然后根据用户角色来分别选择各个角色对于各个表单功能的控制权限。如下图, 管理员选择左侧系统功能列表中的各个子模块, 对于模块中的表单的功能以选择复选框的方式分配用户角色权限。流程步骤中权限分配效果图如下图5-3:



图 5-3 角色权限分配效果图

(4) 系统主界面

通过以上对系统业务活动流程的定制,用户可以进入配件管理系统进行配件管理的业务操作,左侧为配件管理系统的功能模块,随着工作流管理工具对业务流程的定制,系统的业务流程相应发生变化,主界面中相对各个模块的表单中功能的执行次序也发生改变。配件管理系统主界面效果图如下:



图 5-4 配件管理系统主界面效果图

通过以上对系统流程的定制和对用户权限的分配,做到了配件管理系统的业务重组,因篇幅原因,以下按业务流程只给出系统中配件销售模块的配件销售订单、销售发票明细和销售出库单的界面效果图。通过流程实例调用的流程模板对流程的配置顺序为:销售订单提交、销售订单审核、销售发票提交、销售发票审核、出库单提交、出库单审核、发货七个步骤。下面将展示定制成功的订单流转的效果图并加以解释。

(5) 销售订单

配件销售的用户首先需要提交销售订单,点击用户名称下拉菜单选择用户,而且可以点击右方查的按钮对用户进行查询,然后选择制单日期。用户填写完表头信息。同样,在下面配件的信息表中选择相应的配件,并且填写数量。在填写完相应的配件以后,点击保存,提交。等待下一步操作人员审核。本单据制单人的姓名和订单编号是自动生成的,不可修改。当订单需要紧急处理时,当操作员操作完毕之后,选取邮件通知,将自动发送邮件给下一个业务人员提示他进行下一步操作。其具体业务流程在配件销售的需求分

析中已经加以介绍，这里不再详述。配件销售订单效果图如下图 5-5：

配件销售订单

文件 帮助

新增 保存 删除 审核 打印 刷新 退出

配件销售订单

客户名称

制单日期

星期日2008-04-0

是否邮件通知

☒

订单编号

业务员

业务员电话

交货地点

客户联系人

客户电话

摘要

旧编号	配件编号	配件名称	规格型号	单位	数量	单价	金额
1							
2							
3							
4							
5							
6							
7							
8							
						0	0.00

主管

制单人

系统管理员

审核

审核日期

图 5-5 配件销售订单效果图

(6) 销售发票

配件订单完成以后，根据源单类型为销售订单开具销售发票，本单据同样需要审核步骤，如下图 5-6 配件销售发票明细效果图。选择源单类型为销售订单以后，可以根据源单号的下拉菜单加以查询。用户填写完表头和表单的内容，等待审核人员进一步审核。

配件销售发票明细

文件 帮助

新增 保存 删除 审核 打印 刷新 退出

配件销售发票明细

源单类型

销售订单

源单号

XSD008001347

是否邮件通知

☒

发票明细编号

客户名称

开票日期

旧编号	配件编号	配件名称	规格	金额
1			XSD008001358	三包中心门市部批发零售
2			XSD008001390	三包中心门市部批发零售
3			XSD008001417	三包中心门市部批发零售
4			XSD008001420	三包中心门市部批发零售
5			XSD008001424	三包中心门市部批发零售
6			XSD008001426	三包中心门市部批发零售
				0

主管

制单人

系统管理员

审核

审核日期

图 5-6 配件销售发票明细效果图

(7) 销售出库单

根据销售发票可以开具销售出库单，并且进行审核和记账。这里同样也可以选择邮件通知，以实现步骤之间的提醒功能。效果图如下图 5-7 所示：

销售出库单

文件 帮助

新增 保存 删除 审核 记账 导出 分库打印 打印 刷新 退出

销售出库单

源单类型 销售发票明细

源单号

是否邮件通知

发货仓库 三销厂三包新件库

制单日期 星期一2009-04-1

出库单编号

收货单位

出库类别 销售出库

摘要

	旧编号	配件编号	配件名称	规格型号	单位	数量	单价	金额
1								
2								
3								
4								
5								
6								
7								
						0		0.00

主管

制单人 系统管理员

审核人

审核日期

记账人

记账日期

图 5-7 销售出库单效果图

总结与展望

论文讨论的配件管理系统是基于 workflow 技术的信息管理系统,它是一家国有大型汽车制造厂内部使用的应用系统。系统采用了 C/S 加 B/S 的模式,通过汽车制造企业售后服务中心内部配件管理系统与 ASP 平台的数据交换对汽车配件进行信息化管理,实现了配件经销商与汽车制造厂配件供应商之间的业务协作,帮助汽车制造企业实现了提高了售后服务的效率和业务协作能力,降低了服务费用。

本文通过深入的调研某汽车制造厂的实际业务需求,分析了其业务管理的组织机构,并以企业组织机构的管理职能为基础,对制造厂售后服务配件管理中各个环节的业务流程进行了详细的需求分析。通过对企业需求的了解,提出了在 workflow 环境下配件管理系统的设计方法和技术实现。本文立足于对企业具体业务流程的研究,效仿 BPEL workflow 技术提出了自己的技术解决方案。概括起来,本文主要做了以下工作:

(1) 从企业现状和技术要求出发,分析和总结了目前售后服务配件管理系统中的不足,指出了企业目前配件管理系统灵活性不强的缺点。提出了以 workflow 技术为解决企业问题的思想。

(2) 详细的阐述了售后服务配件管理过程中的业务需求和非功能需求,分析了系统需要满足的业务功能需求和非功能需求,借助 UML 建立了系统各个业务模块的流程图模型和用例模型。解决了系统协同性和灵活性的要求。

(3) 通过探索基于 workflow 的配件管理系统的业务目标,分析了解决问题的关键因素,对系统进行了总体设计。针对企业在业务重组(BPR)和业务协同性中面临的问题,将 workflow、数据交换等当前流行的技术有机的结合在一起,以 BPEL workflow 技术的思想为指导,设计了系统的轻量级解决方案。通过对 workflow 基本概念和特征的研究,分析了配件管理系统中的流程特征、人机交互和系统流程的执行过程。并以此为基础,对系统中流程管理和配件管理功能进行了详细的设计。

(4) 本文最后研究并分析了系统的概念数据模型和逻辑数据模型,对数据表的逻辑关系进行了设计,描述了系统中存储过程的使用。论文的实现部分给出了系统的数据库访问实现、系统整体效果的实现以及系统的流程管理模块部分核心代码的实现,最后给出了系统的部分功能实现效果图。

通过这段时间对 workflow 技术的资料收集,理论研究、系统设计、以及最后论文的完成工作。让我对 workflow 技术有了更深的认识和了解,用 workflow 技术实现对企业业务流程的重组,体现的不仅仅是一种技术方式的运用,更有一种业务管理思想融合在其中。从企业的角度去考虑,从业务需求出发,从需求上进行创新,找寻企业最适合的解决方案,才能让 workflow 管理系统物尽

其用。由于时间的仓促以及作者水平所限，本文对基于工作流的配件管理系统的研究在以下几个方面还有所欠缺，将作为下一步研究的重点：

(1) 系统的流程定义模板管理还处于对系统中各个活动接口的挂载，不能实现界面的拖拽，有待以后了解图形化编程的方式来实现。

(2) 系统利用数据挖掘去辅助决策方面能力不足，对企业级信息管理系统来说，对企业数据进行分析、提取，为企业提供决策判断，无疑会大大提升企业的管理质量和管理效率。今后的工作中要进行这方面的尝试去解决数据挖掘，决策支持的问题。

(3) 系统各方面功能还需要进一步完善，如系统的并发流程的控制，系统中业务流程扩展的管理，这些都是针对本系统将来要加以研究的方向。

基于工作流的配件管理系统有待进一步完善，在理论与实践的结合中进步。限于作者水平，文章难免有不足之处，本人将在今后的工作和学习中继续努力，希望自己的研究能为企业信息化发展尽一份力量。

致谢

时光荏苒，近三年的研究生生活马上就要结束了。我诚挚的感谢我的导师孙林夫教授。

在研究生学习阶段我得到了孙老师的悉心指导和淳淳教诲。孙老师一丝不苟的治学态度，严于律己、宽以待人的处世哲学使我在生活和学习上都受益匪浅。正是在孙老师耐心细致的指点和教导下才使这篇论文得以顺利完成。

在毕业设计期间，我还得到了周荣辉老师自始至终的悉心指导，不断加以鼓励，指导我设计的方向。另外，其渊博的知识、严谨的治学态度、不断开拓的创新精神常常令我感动，使我在以后的学习和工作中也必将受益一生。在此，我衷心的感谢周荣辉老师对我的教诲与帮助。同时，我也非常感谢赵慧娟教授、唐慧佳教授、王淑营老师、韩敏老师的淳淳教导和大力帮助，他们无论在生活上还是在学习上都给予了我很大的帮助，正是他们无私的指导和关怀，我的科研和论文工作才能顺利完成，在此向各位老师表示衷心的感谢。

最后，我也尤其感谢我的父母。感谢他们对我一直以来无私的抚养与培养，是他们的汗水滋养我慢慢长大，这种最伟大的爱以及他们对我所做的一切是我一生都无以回报的。

真诚的感谢所有给予我关注的人们。

参考文献

- [1] 李永钧.从“卖汽车”到“买服务”的跨越.上海工业.2004.11, 17~18
- [2] 梁国锋.对轿车售后服务质量的再认识.汽车与配件.2000.4, 314~315
- [3] 唐辉宇.不容忽视的汽车售后服务.汽车工业研究.2003.5, 14~16
- [4] Workflow Management Coalition.The Workflow Reference Model.Technical Report,WFMCTC00-1003,Workflow Management Coalition (<http://www.wfmc.org>). 2009.3
- [5] 王坚, 戴毅茹, 凌卫青. 柔性业务过程管理体系与方法.电子工业出版社.2008.5, 1~33
- [6] 黄波.网络化售后服务管理中的配件管理系统的设计与实现.西南交通大学.2006.4
- [7] 田忠和, 肖梅.基于动态装配式的柔性 workflow 管理系统.计算机应用[J], 2004 vol.24(5), 123~125
- [8] 刘士军, 孟祥旭, 龚斌.支持工程变更的柔性 workflow 系统建模与实现.计算机集成制造系统.CIMS[J],2003.第 9 卷专刊, 79~84
- [9] 余阳, 汤庸.柔性 workflow 元模型及应用.中山大学学报(自然科学版)[J]2004 vol43(2), 21~24
- [10] 褚四斌, 刘金柱. BPM:为业务流程管理提速. 中国计算机用户.2009.1
- [11] 孔建寿. 面向协同产品开发过程的集成管理技术研究. 南京理工大学博士学位论文.2004
- [12] 赵慧娟, 王淑营, 孙林夫.面向中小企业信息化建设的 ASP 服务平台.计算机集成制造系统.2004
- [13] 史鹏翔.汽车网络化销售管理系统的研究与实现.西南交通大学硕士学位论文.2006.5
- [14] 汽车产业链平台设计报告.四川省制造业研究院.2005, 5~10
- [15] 付春岭, 徐京. WS-BPEL 的性能分析与测试. 计算机工程与设计.2009.5
- [16] 梁爱虎.SOA 思想、技术与系统集成应用详解.电子工业出版社.2007.12, 140~155
- [17] 王楠, 刘心雄, 陈和平.Web service 技术研究.计算机与数字工程.2006.7
- [18] MSSQL 入门基础:SQLServer 安全性管理的途径 <http://tech.ddvip.com/2007-01/116783288417289.html>.2009.4

-
- [19] (美) 马利克 著, 汤涛, 邵晓翠 译. ADO.NET2.0 高级程序设计. 人民邮电出版社. 2007.6, 3~37
- [20] 吴朝晖, 邓水光. workflow 系统设计与关键实现. 浙江大学出版社. 2006.12, 1~15
- [21] Kradolf, Geppert A. Dynamic workflow schema evolution based on workflow type versioning and workflow migration[C]. Edinburgh, Scotland: Proceedings of the 4th IFCIS International Conference on Cooperative Information Systems, IEEE Computer Society. 1999
- [22] 梁红洲. 基于 workflow 的人力资源管理系统(HRMS)的研究与设计. 西南交通大学硕士学位论文. 2007.5
- [23] Alonso, D. Agrawal, A. El Abbadi, C. Mohan. Functionality and Limitation of Current workflow Management Systems[J]. IBM Technical Report, IBM. 1997
- [24] 范玉顺. workflow 管理技术基础. 清华大学出版社. 2001.7, 36~38
- [25] R. Tagg et al. Preliminary Design of a Lightweight Workflow Server[M]. Btu Australasian conference Information System, Australia. 1997
- [26] 萨师煊, 王珊. 数据库系统概论. 高等教育出版社. 2004.1, 13~30
- [27] 陶宏才. 数据库原理及设计. 清华大学出版社. 2004.2, 89~91
- [28] 张燕. 基于 CS 和 BS 混合模式的车间生产管理信息系统. 四川大学硕士学位论文. 2005
- [29] 刘唯一, 田雯. 数据模型. 科学出版社. 2001, 17~33
- [30] James R. Groff Paul N. Weinberg 著, 章小莉, 宁欣, 汪永好等译. SQL 完全手册. 2006.6, 450~487
- [31] Mary Pyefinch 著, 希望图书创作室译. SQL 数据库开发从入门到精通. 北京希望电子出版社. 2000.1, 229~248
- [32] 李劲, 谢兆阳. SQLserver2000 数据库设计与系统管理. 清华大学出版社. 2001, 17~23
- [33] 冯玉才. 数据库系统基础. 华中理工大学出版社. 2001, 35~39
- [34] .Net 开发平台 <http://www.ioffice.net.cn/download/tech/5.pdf>. 2009.5
- [35] Sharp, John. Microsoft Visual C# 2005 Step by Step(Pap). Microsoft Pr 2005.11, 1~47
- [36] 岳小婷. ASP.NET 2.0 单点登录技术在数字化校园中的应用. 计算机与现代化. 2009.5
- [37] Gregor v. Bochmann. A UML profile for modeling workflow and business
-

process.2000,8

[38] Marlon Dumas,Arthur H.M ter Hofstede.UML Activity Diagrams as a Workflow Specification Language.Procedings of the UML'2001 Conference[J],2001

[39] 轩兴涛. 基于 PowerDesigner 模型驱动机制下的全程建模研究.西安石油大学学报(自然科学版).2008.6

[40] 华却才让. 基于 PowerDesigner 的大型数据库开发研究和应用. 青海师范大学民族师范学院学报.2008.2

[41] 刘红玉,杜清运,蔡忠亮. 基于 PowerDesigner 的空间数据库建库技术.测绘信息与工程.2007.3

[42] sqlhelper 使用指南

<http://www.host01.com/article/Net/00020001/0561614360959171.htm>, 2009.5

[43] Alvin J. Bruney .Programming Excel Services. Oversea Publishing House.2007.6

[44] 姚从磊.基于 XML 的工作流系统的设计与实现.哈尔滨工程大学硕士学位论文.2004.1

[45] Eric Jung, Andrei Cioroianu, Dave Writz, Mohammad Akif, Steven Brodhead,James Hart.Java XML Programmer's Reference. Wrox Press ;1st edition.2001.6

[46] Yang Dong,Wen Quan,Zhang Shenshen.Modeling workflow using XML and Petri net.Journal of Systems Engineering and Electronics.2004.3

[47]Samue Robert Collins,Shamkant Navathe,XML schema mappings for heterogeneous database access.Information and Software Technology.2002

[48] 李发英. 基于 DOM 与 SAX 的数据存取技术研究 with 实现.信息技术.2009.2

[49]E.Gamma,R.Helm,R.Johnson,and J.Vlissides.Design Patterns-Elements of Reusable Object-Oriented Software.Addison-Wesley.1995

[50] 阎宏.Java 与模式.电子工业出版社.2007.3, 184~190

附录

1.以表单中的“修改”功能为例，以下简略给出用户的权限控制的核心算法

```
public partial class ToolBar : UserControl    //定义用户控件基本工具栏
{
    private bool _modifyVisible = false;
    public bool ModifyVisible    //修改功能的可视化操作
    {
        get
        {
            return _modifyVisible;
        }
        set
        {
            _modifyVisible = value;
            toolModify.Visible = _modifyVisible;
            SepModify.Visible = _modifyVisible;
            itemModify.Visible = _modifyVisible;
        }
    }

    private bool _modifyEnabled = false;
    public bool ModifyEnabled    //修改功能的访问操作
    {
        get
        {
            return _modifyEnabled;
        }
        set
        {
            _modifyEnabled = value;
            toolModify.Enabled = _modifyEnabled;
            itemModify.Enabled = _modifyEnabled;
        }
    }

    public event System.EventHandler ToolModify_Click;    //定义事件
    private void toolModify_Click(object sender, EventArgs e)    //“修改”点击事件
    {
```

```
        if (ToolModify_Click != null)
        {
            ToolModify_Click(this, new EventArgs());
        }
    }
}

// baseForm 类
public partial class baseForm : System.Windows.Forms.Form
{
    public baseForm()
    { InitializeComponent(); //初始化组件 }
    //控制工具条按钮是否显示
    protected void AuthorityControl(UserCtrl.ToolBar toolbar, string FuncName) //
    调用ToolBar
    {
        WinDAL.SysAdminDAL.DataAdminDAL objData = new
        WinDAL.SysAdminDAL.DataAdminDAL();
        String EmpGUID = WinCommon.UserInfo.EmpGUID; //用户Guid
        DataTable rightDT = objData.GetUserFuncAuth(EmpGUID, FuncName); //得到
        用户权限
        toolbar.TableRight = rightDT;
    }
}

// ToolbarForm 继承 baseForm 类
public partial class ToolbarForm : baseForm
{
    protected void toolBar1_ToolModify_Click(object sender, EventArgs e)
    {
        ModifyClick (); //按钮事件例子: “修改”按钮
    }
    protected virtual void AddClick()
    { }
    ..... //定义其他按钮事件
}

//销售出库单 UI 类继承 ToolbarForm
public partial class XFrmSell : BasePage.ToolbarForm
{
```

```
readonly string FormName = "销售出库单";  
public XFrmSell()  
{  
    InitializeComponent();  
    AuthorityControl(toolBar1, FormName); //权限控制、判断用户权限  
}  
}
```

2. 创建一个新的 XML 文档，并将流程数据编译后插入，这里给出部分函数体
/// 创建一个 xml 文件，包括根元素

```
Public void NewXmlCreate (string fileName, string rootName)  
{ XmlDocument doc = new XmlDocument();  
    // 创建一个新的 XmlDeclaration，版本 1.0，编码 gb2312  
    XmlDeclaration newHead = doc.CreateXmlDeclaration("1.0", "gb2312", null);  
    doc.AppendChild(newHead); // 将指定的节点添加到该节点的子节点列表末尾  
    XmlElement newRoot = doc.CreateElement(rootName); // 创建一个根元素  
    doc.AppendChild(newRoot);  
    doc.Save(fileName); // xml 文件写入到磁盘  
}  
/// 添加 xml 元素，包括该元素的子元素和属性。  
public void InsertXml(string fileName, string ModelId, string ModelName, string  
CreatTime, string TypeName, DataTable dt) // file 为 XML 的路径加文件名  
{  
    XmlDocument doc = new XmlDocument();  
    doc.Load(fileName);  
    XmlElement newNode = doc.CreateElement("WFProcess"); // 创建一个新元素  
    newNode.SetAttribute("id", ModelId); // 创建该元素的属性，模板 id  
    newNode.SetAttribute("name", ModelName); // 模板名  
    XmlElement ProcHeader = doc.CreateElement("ProcHeader"); // 创建该元素的  
    子元素  
    newNode.AppendChild(ProcHeader);  
    XmlElement Created = doc.CreateElement("Created"); // 创建时间  
    Created.InnerText = CreatTime;  
    ProcHeader.AppendChild(Created);  
    XmlElement ModelType = doc.CreateElement("ModelType"); // 模板类型  
    ModelType.InnerText = TypeName;  
    ProcHeader.AppendChild(ModelType);  
    for (int i = 0; i++; i < dt.Rows.Count)  
    {
```

```
ActInsert(dt.Rows[i][0].ToString(),dt.Rows[i][1].ToString(),dt.Rows[i][2].ToString(),dt.Rows[i][3].ToString(),dt.Rows[i][4].ToString());
}
doc.DocumentElement.AppendChild(newNode);
//xml文件写入到磁盘
XmlTextWriter xtw=new XmlTextWriter(file,null);
xtw.Formatting=Formatting.Indented;
doc.WriteContentTo(xtw);
xtw.Close();
}
public void ActInsert (string activeId,string activeName ,string condition, string
roleId,string roleType)//插入活动
{
XmlDocument doc=new XmlDocument();
XmlElement newNode = doc.CreateElement("Activity");
//创建该元素的属性
newNode.SetAttribute("id", activeId);
newNode.SetAttribute("name",activeName);
CondInsert(condition, roleId, roleType );//插入跳转条件
doc.DocumentElement.AppendChild(newNode);
}
public void CondInsert(string condition, string roleId,string roleType )//插入跳
转条件
{
XmlDocument doc=new XmlDocument();
XmlElement newNode = doc.CreateElement("ActPerformer");
newNode.SetAttribute("Performer", condition); //创建该元素的属性
RoleInsert(roleId, roleType)//插入角色控制
doc.DocumentElement.AppendChild(newNode);
}
Public void RoleInsert(string roleId,string roleType)//插入角色控制
{
XmlDocument doc=new XmlDocument();
XmlElement newNode = doc.CreateElement("Role ");
newNode.SetAttribute("Id", roleId); //创建该元素的属性
newNode.SetAttribute("Type ", roleType);
doc.DocumentElement.AppendChild(newNode);
}
```

//解析 XML 函数

```
private XmlDocument doc= new XmlDocument();//实例化XmlDocument
```

```
private Hashtable ht = new Hashtable();
```

//该方法采用了递归调用的方式，方法体有2个参数

```
private void LoadXmlElements(TreeNodeCollection treeElements, XmlNode  
xmlNode)
```

```
{ switch (xmlNode.NodeType)//判断节点类型
```

```
{
```

```
case XmlNodeType.Element://这里只查找节点元素的值，如<item>
```

```
TreeNode treeNode = new TreeNode(xmlNode.Name);
```

```
treeElements.Add(treeElements);
```

```
ht.Add(treeElements, xmlNode);//将节点放入哈希表
```

```
foreach (XmlAttribute xmlAttr in xmlNode.Attributes)//遍历节点属性
```

```
{
```

```
TreeNode attrNode = new TreeNode("@ " + xmlAttr.Name);
```

```
treeElements.Nodes.Add(attrNode);
```

```
ht.Add(attrNode, xmlAttr);
```

```
}
```

```
foreach (XmlNode xn in xmlNode.ChildNodes)//遍历子节点
```

```
{
```

```
LoadXmlElements (treeElements.Nodes, xn); //递归调用
```

```
}
```

```
break;
```

```
case XmlNodeType.Entity:
```

```
.....//如果是节点实体<!Entity>
```

```
break;
```

```
case XmlNodeType.Text:
```

```
.....//节点的文本内容
```

```
break;
```

```
case XmlNodeType.Comment://节点注释
```

```
break;
```

```
default:
```

```
break;
```

```
}
```

```
}
```

```
public string selectElementByTag(string filePath, string TagName)//file为文档  
路径TagName为xml标签名
```

```
{//根据xml标签查找文档中元素
```

```
string return nodeString= "";
XmlDocument doc = new XmlDocument();
doc.Load(filePath);
XmlNode serchNode = doc.SelectSingleNode(strSearch);//查找与strSearch匹配的节点
if (serchNode != null)
{
return nodeString = serchNode.InnerText;
}
return return nodeString;//返回节点字符串
}
```

攻读硕士学位期间发表的论文及科研成果

- [1] 崔勇杰, 李斌勇, 黄彦琦. 基于工作流的汽车配件管理系统的研究. 《电脑与信息技术》2009.6