

中国计算机技术服务公司
全国计算机技术培训网统编教材

汉化 dBASEⅢ 管理软件 分析与设计

李一智 主编
高 阳

清华大学出版社

汉化 dBASE III 管理软件 分析与设计

李一智
高阳
陈晓红
向文光

(主编)

吴良刚
罗新星

编著

清华大学出版社

内容简介

本书的内容是阐述企(事)业管理软件系统分析与设计的方法和通用案例,其环境系统亦为国内流行的 IBM-PC/XT、长城系列机或其兼容机与 C-dBASE III 和 BASIC 语言。本书共分十章:第一、二章论述管理软件开发的原理、方法、工具与常见问题的解决技巧;第三章介绍开发了 25 个通用模块程序系统;第四章至第十章分别以案例形式介绍日、月统计报表、物资供应、销售(含合同)、财务会计、劳动人事、职工安全以及产品质量等管理信息系统的分析与设计及若干模块程序清单。本书另外有 8 个管理系统全套程序软盘。

本书可作为高校与中专经济、管理类有关专业管理信息系统、dBASE III 数据库技术、软件开发技术等课程的教材或案例教学参考书,也可作为企(事)业有关技术、管理人员的培训教材或参考书。

汉化 dBASE III 管理软件分析与设计

李一智等 编著

☆

清华大学出版社出版

北京 清华园

国防工业出版社印刷厂印装

新华书店总店科技发行所发行

☆

开本: 787×1092 1/16 印张: 21 1/8 字数: 575 千字

1990 年 9 月第 1 版 1991 年 8 月第 2 次印刷

印数: 5001—15000

ISBN 7-302-00820-5/TP·296

定价: 13.00 元

出版说明

在世界新技术革命中,计算机已成为一个崭新的力量,成为最活跃、最先进的核心技术之一,在信息社会中发挥着她的强大威力。要使我国计算机应用事业尽快地赶上世界先进水平,人才的培养是十分重要的。机械电子部计算机技术培训中心、中国计算机技术服务公司技术培训网和中国计算机学会技术培训学组担负着在全国范围内对计算机应用人才进行培养的重任。

为了能迅速、有效地提高计算机技术培训的质量,使技术培训向正规化、系列化、分层次方面发展;为在我国建立一支宏大的应用计算机的队伍,机械电子部计算机技术培训中心、中国计算机技术服务公司技术培训网和中国计算机学会技术培训学组组织了培训网系统内的培训中心、培训部门及部分高等院校、科研院所、计算机生产厂等单位的计算机专家组成了全国计算机技术培训网教材编审委员会。教材编审委员会从国内外计算机技术发展和我国实际情况出发,会同北京地区六家出版社,经过有计划地选题、编写大纲、大纲审定、指定主审和主编、具体编写、审稿、编辑、出版、印刷、发行等整个环节在全网范围内已经编写教材百余种。这些教材的主要对象是非计算机专业(包括从事计算机应用的计算机专业)的广大科技人员和管理人员,也可以作为高等院校的教学参考书及大专院校学生和从事计算机应用人员的自学教材。这些教材本着两个指导思想进行编写,即实用性强;让读者学完后能立即用上;跟踪新技术、新成果、新趋势快;让读者及时掌握最先进的技术服务于社会。在培训工作方面遵循三条宗旨,即面向全国、面向应用、面向用户,为读者用好计算机服务。

为进一步推动计算机应用事业的发展,在国家教委和机械电子部有关部门的支持下制定了“计算机应用人才培训规范”,并已在机电行业试行。规范明确了对计算机应用人才的培养目标、知识、能力和水平的要求,以及为达到目标和要求而设置的课程。课程按操作员、程序员、高级程序员和系统分析员等四个层次及数据处理、检测与控制、CAD/CAM等三个应用领域,并同时考虑全国软件人员水平考试的需要进行设置。规范所包括的57本教材将由全国计算机技术培训网教材编审委员会组织有实践经验的专家、教授编写,并于89年开始陆续发行。

全国计算机技术培训网教材编审委员会

75366/05

编审委员会名单

名誉主任
副主任
委员

陈力为
邵祖英
黄安南

(按姓笔划为序)

于占涛
李克洪
张宇铭
徐国平
邓小敏

王春元
何积功
金锡智
夏涛

任仲贵
杨德源
林哲生

李潮义
杨学良
钟圣雷

秘 书

前 言

我国企业管理现代化,是在改革、开放政策和系统管理思想的指导下,建立科学的、高效率的管理组织与制度,采用先进合理的管理方法,使用现代化管理手段——电子计算机与现代通讯技术,以期提高企业管理水平和经济效益,适应社会主义生产力发展的需要。

计算机辅助管理,本身既是一种现代化手段,又是促进管理方法现代化、决策科学化的重要工具。我国近十年来,工业企业应用计算机辅助管理发展迅速,有的已经建立全企业的管理信息系统,甚至局部地发挥了决策支持作用。但是有关会议讨论、总结了十多年来计算机辅助管理的经验教训,许多专家感到,由于多数企业目前的管理基础工作还不够健全、技术力量和条件不足等主客观原因,致使仓促建成的管理信息系统,未能发挥应有的功能。因此认为,计算机辅助管理必须稳步前进,当前还应扎扎实实地在系统总体设计指导下,使用微机实现各项业务的信息处理,促进管理水平的提高;在此基础上逐步过渡到全面的管理信息系统(含决策支持)以及网络化阶段。按照这种意见,企业各个业务部门的管理信息系统软件的开发——分析与设计,将为多数企业更加重视。本书就是为适应上述环境要求而编著的,也是编著者多年来承担的科研项目成果的总结。

本书的内容是企(事)业管理软件系统的分析与设计。考虑到工业企业类型较多,业务区别较大,但又要尽量达到介绍的管理软件通用化,故在第三章专门编写了管理软件开发中具有共性的25个通用模块程序系统,它将成为任何企(事)业、不同部门开发管理软件时借鉴的有效工具,或参考调用的程序成分;此外,第四至十章,精选了工业企业中比较具有共性的七个部门管理系统,分别系统地介绍了管理软件开发的系统分析、系统设计和程序设计,以及具有特色的模块程序清单。至于生产计划管理部门等的软件,由于业务内容各企业差别十分大,很难通用化,故未编入;但其中稍有共性的部分,如计划统计日、月报表的信息处理等,已经编入第四章。第一、二两章论述了管理软件开发的原理、方法与工具,以及应用dBASE III程序设计常见的若干问题及其解决技巧。

为了节约篇幅,又想让读者全面掌握若干部门管理软件系统的全貌,本书另外有第三章至第十章的八个系统的全套程序软磁盘,软件开发环境为IBM-PC/XT、长城系列机及其兼容机与C-dBASE III和BASIC语言,各系统皆采用人-机对话和菜单选择方式,还包含试范运行所需的数据库文件。读者阅读各章内容以后,即可方便地使用相应的系统软件,以利开发自己企业的管理软件。亦即本书编写的目的是注重管理软件开发的实践,希望能够推广使用所附的管理软件。

为了阅读方便,由简到繁循序渐进,建议阅读各章的顺序为第一、二、三、九、八、四、五、六、十、七章;无论如何希望读者先应基本掌握BASIC语言和dBASE III语言,这些语言已有专著介绍,本书仅对后者在程序设计和技巧上作了介绍。

本书由中南工业大学李一智(第一、九章)、高阳(第一、二、三、四章)、陈晓红(第三、七章)、吴良刚(第五、八章)、向文光(第六、十章)和罗新星(第四章部分)共同编著。由李一智教授、高阳副教授负责主编定稿。由清华大学高云鹏副教授主审。本书另外所附的全套软磁盘的管理软件由陈晓红、吴良刚和向文光最后负责总调试。

本书的出版得到清华大学出版社和中国计算机技术服务公司的负责同志的大力支持和关

怀,于此表示谢意。还要感谢中南工业大学管理工程系的领导和有关同志的支持。对研究生黄建军、何晓洁和袁剑以及本科生张一燕等同志所给予的帮助,表示谢意。

由于编著者水平有限,疏误之处在所难免,恳切希望读者指正。

编著者 1989年9月

目 录

前言

第一章 管理软件开发基础	1
§ 1.1 管理软件概述	1
§ 1.2 管理软件开发方法学	3
§ 1.3 系统分析	8
§ 1.4 系统设计和编码	10
§ 1.5 程序设计风格	16
§ 1.6 软件测试	18
§ 1.7 文档种类与编制	21
§ 1.8 软件管理	24
§ 1.9 dBASE III 概述	27
第二章 dBASE III 程序设计的若干问题	33
§ 2.1 CONFIG.DB 文件及其应用	33
§ 2.2 循环及其应用	37
§ 2.3 宏代换函数及其应用	40
§ 2.4 数组的生成与应用	42
§ 2.5 提高 dBASE III 的运行速度	44
§ 2.6 程序加密	48
§ 2.7 清除存贮空间的有效占用	50
§ 2.8 报表打印的一些问题	50
第三章 通用程序系统	54
§ 3.1 主控模块和公共子模块	54
§ 3.2 建立数据库	65
§ 3.3 数据输入和增删修改	69
§ 3.4 数据库组织维护	83
§ 3.5 数据分类	88
§ 3.6 检索统计	93
§ 3.7 关联操作	105
§ 3.8 库间计算及库间编辑	115
§ 3.9 百分比计算	122
§ 3.10 文件管理	125
§ 3.11 报表打印通用模块	130
§ 3.12 报告及标签打印	143
§ 3.13 定义功能键和外部接口	149
第四章 日、月报表信息处理系统	153
§ 4.1 系统简要分析	153
§ 4.2 系统设计	155
§ 4.3 程序设计	160
§ 4.4 系统评价	173
第五章 企业物资供应管理信息系统	176
§ 5.1 系统分析	176

§ 5.2 系统设计	181
§ 5.3 程序设计	193
第六章 销售管理信息系统	204
§ 6.1 系统分析	204
§ 6.2 系统设计	207
§ 6.3 程序设计	211
第七章 财务管理信息系统	226
§ 7.1 财务管理信息系统概述	226
§ 7.2 财务管理的基本内容	226
§ 7.3 系统分析	229
§ 7.4 系统设计	232
§ 7.5 程序设计	241
§ 7.6 系统评价	255
第八章 企业劳动人事管理信息系统	256
§ 8.1 系统分析	256
§ 8.2 系统设计	258
§ 8.3 程序设计	268
第九章 厂矿企业职工安全管理系统	285
§ 9.1 系统分析	285
§ 9.2 系统设计	287
§ 9.3 程序设计	291
第十章 产品质量管理系统	296
§ 10.1 系统分析	296
§ 10.2 系统设计	300
§ 10.3 程序设计	304

第一章 管理软件开发基础

计算机专家 Larry E. Druffel 说：“软件将是本世纪社会中最重要技术之一。我们不能仅仅编写程序，我们必须懂得设计和管理软件系统”。为此，本章将着重讨论企事业单位管理软件的分析、设计、实现、测试的有关原理、方法、技术和工具，主要包括管理软件开发方法学、系统分析、系统设计、编码以及软件测试和软件管理；同时，由于目前国内主要以 dBASE III 为工具来开发管理软件，因此本章还对 dBASE III 进行较全面的概述。无疑，这些内容都是目前管理软件开发的基础。

§ 1.1 管理软件概述

在讨论软件开发之前，有必要对软件的含义、类别、特点及其发展历史作简要的介绍。

一、软件及其类别

今天，“软件”与“系统”、“信息”等词一样，已广泛使用于科技界以至日常生活之中。平常，有人把方法、策略、措施等等称为软件，类似地也有“软科学”一词。显然，软件相对于“硬件”，即客观存在的实体而言，具有“软”的特性，即所谓非实物性。一般地讲，软件是智力产品，具有抽象性与严密的逻辑性。在数据处理领域，硬件指数据处理中使用的物理设备，如计算机及其外部设备。软件指与计算机操作有关的程序以及开发、使用和维护程序所需要的各种文档，由此，软件也可以说是程序、数据与文档的集合。而文档往往是人们所不重视的，但在软件工程中，文档的建立却是极为重要的工作，务必引起充分的注意。

计算机软件分为系统软件与应用软件两类。系统软件是为专门的计算机系统或一族计算机系统所设计的，用来支持计算机系统及有关程序的操作和维护。例如操作系统、编译程序、诊断程序等都是系统软件。它一般由计算机制造厂家提供，购机时可以选购或随机携带。应用软件是使一台计算机得到某种稳定功能而专门研制的软件。它往往由计算机用户根据所要解决的问题自己编制，多用高级语言编写。随着计算机的普及和广泛使用，应用软件的种类越来越多。例如，有用于导航、火炮控制、工程设计、科学计算、过程控制、企事业管理、人工智能等方面的应用软件。本书所说的管理软件就是为解决企事业信息管理而研制的一类应用软件。

二、管理软件的特点

在目前的计算机系统中，软件已经取代硬件而成为最难设计，最小可能成功，最难管理的系统成分，这与软件的特点密切相关。因此，下面简要讨论一下软件的主要特点。

· 软件的非实物性

软件是智力产品，是人们逻辑思维的结果，用文档和程序表达出来的。因此，从形式上来说，它是符号、文字与图表的集合，是非实物型的产品，可见性差。这使得软件难以开发、维护和管理，也使得软件的质量难以衡量，从而要求软件开发人员具有较高的素质与技术水平。此外，软件的非实物性，也不易使人们看到它的价值，以致不愿意花钱去买。

· 软件产品的一次性

软件开发是典型的单一产品生产，每一个软件的研制都是一个新开发过程，即一次性开

发。软件复制显然不能视为软件开发。由于每次都是新的开发,因此就影响了软件的生产率。

- 软件不会用旧

任何实物型产品,如一台电视机,经长期使用后,都有磨损或老化的问题。但软件与此不同,它不会用旧。比如一个工资系统,如果用户的要求没有变化的话,则它无论运行多少年都无所谓新旧的问题。因此,软件的故障不能用普通产品更换零件的方法来解决,而只能修改。尽管软件无所谓新旧的问题,但由于外部环境的变化、用户要求的变化以及软件本身含有的错误和缺陷,所以软件与人一样,仍然有生命周期的问题。

- 软件研制容易出错

一般工程产品,都有严格的质量标准,并且在制造过程中,要随时进行检查,大部分均可定量检测,因而可以保证产品质量。软件是智力产品,没有明显的制造阶段,目前还难以对它的质量进行量化的全面度量,自然就难以保证其质量。软件测试只能发现错误,而不能保证没有错误。证明程序正确性的研究虽已取得了一定成果,但离实用相差甚远。这样,程序在运行过程中,还可能会不断出现错误。另外,外界环境以及用户的要求也在不断变化,因而对软件要不断地进行维护,包括修改程序,甚至改进设计。

- 软件容易复制

软件开发是既费钱又费时间的工作,大型软件需要数百人年甚至数千人年的工作量。然而,软件的复制却是轻而易举的。因此,有人担心软件成果容易转为别人所有,而不愿意开发或不愿意将软件投入市场,这样也就影响了软件市场的发展。

管理软件除具有上述所有的特点之外,还由于它是为企事业管理服务的,业务繁琐,数据量大,因此应合理地组织数据。此外,管理软件的运行环境是管理系统,它的输入来自管理,输出也是为了管理。

三、管理软件的发展阶段

管理信息系统由管理系统、信息处理系统以及数据传输系统所组成,而管理软件是信息处理系统的核心部分。因此,管理软件随管理信息系统的发展而发展。迄今,在发达国家中,管理信息系统的发展已经历了萌芽、起飞和成熟三个阶段。

自 1945 年世界上出现第一台计算机到 60 年代中期,是管理信息系统发展的第一个阶段,即萌芽阶段。在这个阶段中,硬件虽然经历了电子管计算机到晶体管计算机的变革,但这时机器仍然昂贵。到 1965 年,美国才安装了大约两万台计算机,其中大部分用于企事业管理。计算机界对于软件的研制和发展不够重视,而主要力量仍然用于研制硬件。这时的软件开发处于程序设计阶段。程序为每个具体应用而专门设计,而且多数是由相同的人员来开发和最终使用。由于这种个体化的软件环境,设计是在人们头脑里进行的隐含过程,除了源程序之外,几乎没有文档资料。这时的系统多数为批处理工作方式,个别为交互式系统。

管理信息系统发展的第二个阶段是从 60 年代中期到 70 年代中期,这是管理信息系统迅速发展的阶段,国外有人称为“起飞”阶段。这时,硬件从晶体管计算机发展为集成电路计算机,CPU 的速度和内存容量大大提高,成本大大下降,从而为计算机的广泛应用提供了条件。发达国家的企事业单位大部分建立了计算机管理信息系统。多道程序设计,多用户系统引进了人机交互的新概念,交互技术开辟了新的应用领域。实时系统能从多个数据源收集、分析和传送数据,联机辅助存储设备的进步导致第一代数据库管理系统的产生。同时,研制软件的“软件作坊”颇为盛行,但依然使用早期的开发方法。从全面来看,此时的软件开发称为程序系统阶段。

由于这个时期电子技术的飞跃发展,硬件成本降低,软件成本相对增加。特别随着软件数量的急剧增加和规模的扩大,如由数万条语句或数十万条语句构成的软件产品日益增多,从而

导致软件的维护以惊人的速度耗费资源。而程序的个体化特性使得它们实际上是不可维护的,加上大规模软件的复杂性,使得其开发难以成功,于是“软件危机”出现了。然而,直到1968年在西德召开的国际软件工程会议上,才普遍认识到危机的存在,并正式提出和使用“软件工程”一词。从此,一门新兴的工程学科就诞生了。

从70年代中期一直到现在是管理信息系统发展的第三个阶段,即成熟阶段。这个阶段,硬件从集成电路计算机发展到超大规模集成电路计算机,微型机的大量普及,各种外部设备的增加以及通讯技术的发展,使管理信息系统向分布式方向发展,出现了连接多个企业的信息网络。发达国家的企事业中,已普遍建立管理信息系统,并逐步与生产过程的监测和控制相结合,利用局部网络实现分布式处理。美国至今已有大中小计算机100万台,微型机1000万台以上,其中90%用于企事业。企业的管理信息系统已发展为综合的经营管理信息系统。这个时期,软件工程获得了重大进展,已相继提出了一系列软件开发的观念、原理、方法、技术和工具,软件开发已进展到软件工程阶段。但是,硬件的高速发展,已经超越了提供支持软件的能力,以致软件供需矛盾进一步恶化,软件成本日益增长,软件质量不能满足系统产品的严格要求,软件生产率的提高远远落后于硬件技术发展的速度和计算机应用领域扩大的步伐。当前,软件维护费用已占数据处理预算费用的50%以上,于是软件危机进一步加剧。因此,如何极大地提高软件的生产率及其质量,是当前国际上软件工程中的紧迫任务。

当前,软件工程的研究和发展主要集中在下述方面:

(1) 发展软件工程支撑环境

其主要思想是把软件工程领域中的各种方法和技术都转变成支撑系统或软件工具,以支持开发人员更加有效地完成特定的软件开发与维护任务。

(2) 采用智能系统

人工智能技术将应用于软件工程中,实现软件开发的自动化,使用户需求规格说明能直接翻译成合适的设计规格说明,而这种设计规格又能自动地转换成程序。

(3) 软件的重复使用

采用能重复使用软件的设计方法学,我们将能使用现有的设计文档、设计规格说明、设计结构、模块、数据等等。如果能将软件成分标准化,那末就可以象机器设备自动组装一样,部件和组装的概念也可用于软件生产。这样就能大大减少开发费用并提高软件质量。因此,今天改进软件生产过程及其产品的最有前途的办法之一是软件的重复使用。

(4) 发展软件维护的理论与工具

目前软件维护费用急剧上升,因此发展软件维护的理论与工具就显得十分重要。预计美国的 Stars 工程,英国的 Alvey 计划,日本的 Σ 系统实现以后,必将促进软件维护理论的新进展,以及新的维护技术、维护工具的产生和应用。

就我国而言,自80年代初期开始广泛使用微型机,但至今仍只有少数企业用微机进行单项管理,真正建成了管理信息系统的为数不多。可见管理信息系统尚处于发展时期,软件开发还未进入软件工程阶段。因此,如何广泛推广和运用软件工程的基本原理、方法、技术和工具就显得格外重要。

§ 1.2 管理软件开发方法学

上节提到,软件危机的一个重要原因就是软件系统的高度复杂,难于驾驭。要解决软件危机,必须应用严格的技术,使用帮助管理并降低其复杂性的工具。为此,我们应该进一步探讨软件开发的目标,研究帮助实现这些目标的简单原理、方法和工具。

一、软件开发目标

软件开发的主要目标如下：

- 满足用户的需求

软件为用户开发和使用,显然,软件开发的首要目标是使软件产品满足用户的要求。尽管人们不难理解,但如何真正做到却并非易事。这既有开发者的原因,也有用户的原因。事实上,很少有一致的或完整的需求规格说明书,用户以及开发人员对问题常有不完整的理解,致使在开发过程中往往改变需求。此外,开发者的素质和水平也是一个重要因素。为此,我们必须有一系列不受这种变化影响的措施,诸如可修改性、有效性、可靠性和可理解性等等。很清楚,它们既是措施,也是软件开发所追求的目标。

- 可修改性

长期运行的软件,由于要响应用户需求的变化,以及在系统分析和设计阶段所引入的错误,故不得不修改。若系统是可修改的,就应该允许进行修改而不致增加原系统的复杂性。为了系统可修改,系统的结构应该模块化,而且模块应与问题或功能相对应。当某个问题或某个功能发生变化时,则只需修改该问题或功能所对应的模块。

- 有效性

有效性指软件系统以最佳方式在使用可用的时间资源和空间资源的情况下正常运行。如果某种处理必须在给定的时间内运行(如能源管理系统中传感器的采样),则必须限制时间资源。我们应该注意的是:“……对于一个错误的结构,对其有效性进行苦思冥想是无济于事的,考察对问题更加一致的理解,对于效率才有更大的影响。”

- 可靠性

可靠性是一个极为重要的目标。对工程产品而言,没有可靠性,就没有使用价值。例如一个电力负荷管理系统的失效,可能导致系统内的所有电厂的破坏。无疑,这种系统是不能存在下去的。因此,开发人员应千方百计从各个不同的角度来提高软件的可靠性,要防止因概念、设计和结构不完善造成的失效,又要能挽回因操作或性能不适造成的失效。

- 可理解性

软件需要修改,一个不可理解的系统,就不能进行正确的修改。因此,可理解性有助于软件的维护和管理。一个可理解的系统,必须直接反映问题的本来面目。对于建成可修改的、有效的和可靠的系统来说,基本的要求是在编码阶段,要注意编码的风格,程序应该是可读的。N·Wirth曾在第五届国际软件工程会议上说:“在软件设计中,我们应该提倡的是简明、朴素、思想清晰、便于理解。”

二、软件工程的原理

为了实现上述目标,必须遵循下列简单而基本的原理:

- 抽象

抽象的概念由来已久,它是如此重要,以致我们做每一件事时都要用到它。如在 COBOL 中,数据结构的层次为数据库、文件、记录、数据项。我们知道,数据库是文件的集合,它是对文件的抽象。同理,文件是对记录的抽象,记录是对数据项的抽象。越往后,抽象的级别越低。可见,抽象的本质是抽取基本的性质而忽略非本质的细节。现实世界中,事物、状态或过程总存在某些相似的方面,当把这些相似的方面集中和概括起来,暂时忽略其差异,这就是抽象。

人类思维能力不是无限的,如果同时面临的因素太多,就不可能做出精确的思维。处理复杂问题唯一有效的方法,是用层次的方式构造和分析它。一个复杂的问题,首先可用一些高级

的抽象概念来构造和理解,这些高级概念又可以用一些较低级的概念构造和理解,如此进行下去,直至最低层的具体元素。

软件开发中的每一步都是在软件解法的抽象级中逐步求精。在系统分析阶段,分层数据流图即为抽象的结果。在系统初步设计阶段,自顶向下,使用分解和抽象,可以得到系统结构图。结构图中的每一层模块,表示了对软件抽象层次的一次精化。当进入到详细设计阶段时,抽象级降低。最后产生源程序时,即达到了抽象的最低层。可见,在系统开发的全过程以至每一阶段、每种分析和设计的方法中都与抽象分不开。

抽象的最大好处不只是有效性,还有其他好处。例如,抽象可以减少程序员在任一抽象层次需要了解的细节,有助于提高系统的可理解性和可维护性。此外,在抽象的每一层次仅允许特定的操作,这样,禁止了任何违反这一层次逻辑观点的操作,也提高了系统的可靠性。因此,应该很好地运用这一有力的思维工具,以便更好地驾驭系统,构造系统。

• 信息隐藏

抽象是抽取给定层的基本内容,其余细节则隐藏起来。隐藏的目的在于使影响系统其它部分的某些细节变得不可访问。当进行系统结构设计时,根据信息隐藏原则,应该这样来设计和确定模块,使得包含在某模块内的信息(过程和数据),对于不需要这些信息的其它模块来说是不能访问的。这样,提高了模块的独立性,从而也即提高了系统的可修改性和可靠性。如果在软件测试和以后需要修改软件时,由于大多数数据和过程对其他软件部分是隐藏的,则因疏忽而引起的错误就很少可能扩散到软件的其他部分。

• 模块化和局部化

模块化是解决管理软件系统复杂性的另一个基本工具。模块化描述系统的结构如何更容易地达到某种目的,所以模块化是目标结构化,目前,几乎所有软件结构化设计技术的基本概念都是以模块的描述为前提的。模块是离散的程序单位,可以单独命名和编址,能独立地设计、编码、调试、修改和维护。在dBASE III中,一个模块即为一个命令文件。模块化的实质是运用系统原理,采用从上到下的推理方法,将系统分解为若干个子系统,子系统还可依次分解下去。系统和子系统都有各自的目标与功能,每一个功能在一个单独的模块中实现。当然在系统的分解过程中也要运用抽象和信息隐藏原理。

如果系统要有较高的可靠性,就必须对每个模块提供经过良好定义的接口。不管一个程序模块怎样被定义,它总要与其他模块相互作用。因此,定义了耦合性和内聚性两个基本概念。耦合性(Coupling)是模块间相互联系紧密程度的度量。理想的是,希望设计松散耦合的模块,这样就能相对独立于其它模块来处理单个模块。内聚性(Cohesion)是模块内诸成分相互联系紧密和相关的程度。我们希望模块有较强的内聚性,使得一个模块内的成分是功能和逻辑相关的。

局部化主要考虑物理方面的相似性,要求在一个模块内集中逻辑相关的计算资源。此外,局部化意味着一个模块应当是充分独立的。这样,它就有松散的耦合结构。可以看出,模块化和局部化原理直接支持可修改性、可靠性和可理解性。一个有良好结构的系统,对每一个模块都容易了解。由于把设计决策局部化在几个模块中,故一次修改只影响很少的一些模块。此外,如果应用了目标模块化,就限制了程序模块间的互相联系,从而增加了程序的可靠性。

• 一致性、完整性和可验证性

抽象和模块化固然是控制软件复杂性最重要的原理。然而,这并不充分,因为他们不能保证最终将建立一致和正确的系统。必须应用一致性、完整性和可验证性的原理,才能保证实现这样的系统。

一致性指各模块使用一致的符号,对操作一致的控制结构和调用序列,在任一抽象层次和逻辑相关对象的表达是相同的。这一般由好的设计风格得到。

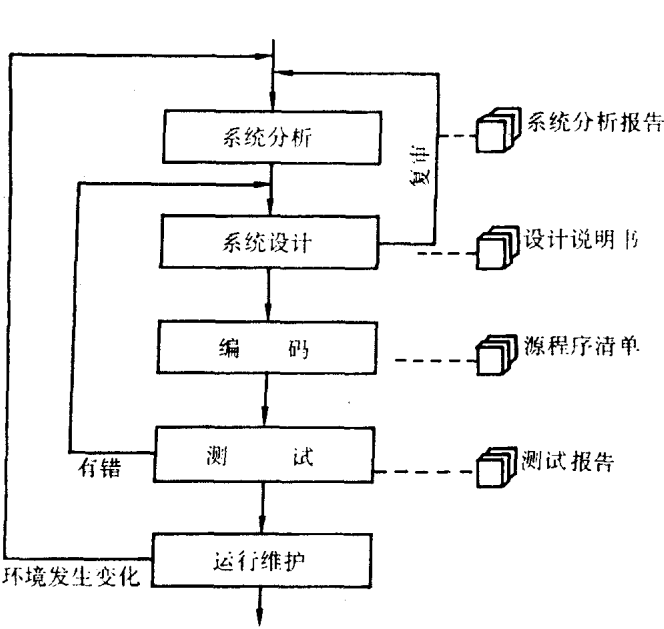
抽象是提取给定实体的本质内容,完整性保证不丢失实体的任何重要成分。在某种意义上,抽象和完整性帮助我们开发必要且充分的模块,亦即正确的解,从而支持可靠性、有效性和可修改性的目标。

可验证性的原理指出,分解系统时必须使它能被容易地测试,从而也可以对系统进行修改。

三、开发方法

从软件开发的全过程来看,目前主要有生命周期法和原型法。在各开发阶段,还有多种具体方法和技术,如结构化分析(SA)、结构化设计(SD)、结构化编码(SP)、结构化分析与设计技术(SADT),基于功能分解的自顶向下设计(TDD),Jackson 结构设计法(MJSD)等等。这里只简要介绍生命周期法和原型法,其他有关具体方法和技术将在后面讨论。

1. 生命周期法



同任何事物都有产生、发展、消亡的过程一样,软件从分析设计开始,到它不能再使用的时间周期称为软件的生命周期。它典型地包括系统分析、设计、编码、测试及运行维护五个阶段。每一个阶段还可细分为若干个步骤。事实上,阶段划分的个数尚无统一的标准,但基本内容是一致的。这种划分是为了有效地管理软件的研制过程而提供的一个轮廓。实际系统的开发工作不可能是直线进行的,常常存在着反复,研制人员往往需要从后面的阶段回溯到前面。例如,在设计阶段发现用户个别要求不明确或功能不清楚,就要回到分析阶段进行再分析。因此,生命周期开发过程如图 1-1 所示。

图 1-1 生命周期法开发过程

软件生命周期的前四个阶段,通常称软件开发期,最后一个阶段称维护期。据统计,一般开发期和维护期分别占生命周期的 33%和 67%。在开发期内,各阶段所占工作量的情况如表 1-1 所示。

表 1-1 各阶段工作量比

阶段名称	系统分析	系统设计	编 码	测 试
工作量比%	20	15	20	45

可见,在软件开发期中,测试的工作量大,而在软件整个生命周期中,维护期长,工作量很大,这些应引起充分注意,以便进行合理的安排和资源分配。

要注意的是,我们会常常碰到许多所谓小项目的软件开发,工作量不大。即使如此,也要按照软件工程的原理、方法、技术和步骤进行开发,只不过有些部分可以简化,有些文档可以合并。但总的来说,软件工程的原则不变。

在国外,生命周期法已成功地开发了许多工程项目。它的主要特点是开发过程阶段清楚,每一步都有明确的成果,这些成果以图、表、文本等形式确定下来。这样就便于整个开发过程的管理和控制。为了进一步地了解它,后面对系统分析、设计、编码、测试等还要进行较详细的讨论。

生命周期法的主要缺点是:仅在分析阶段,用户和系统分析员共同合作,而以后的工作一般由开发人员独立完成。而在分析阶段,要确定系统各部分的功能,在许多情况下是困难的。用户限于对计算机知识和应用系统缺乏了解,往往只能大致提出自己的要求,而当系统实现后又感到不满意,需要修改。尤其对于大型软件系统来说,用生命周期法来开发,一般需要2~3年。这样当系统实现时,原来提出的要求往往有了很大的变化,这就使得新系统很快就不适用了。于是,近年来出现了一种新的方法——原型法。

2. 原型法

原型法(Prototyping Approach)的原理类似于工程设计中长期使用的制造模型法。例如,要造一台新的万吨水压机,一般先造一个模型,对模型进行反复讨论修改以得到最终设计方案,然后真正动手制造。与此类似,软件原型的设计是个反复迭代的过程。最初的原型是非常简单的,它只能实现系统一些最基本的功能。随着不断试验、纠错、使用、评价和修改,获得新的原型版本。这一过程周而复始。通过增量方式。每一次修改后的原型版本将完成更多的任务,具有更强的功能,能更进一步满足用户的要求。最终,软件原型可能成为真正的产品。可见,所谓原型是一个可以修改,可以实际运行并不断完善系统。

原型法开发过程的框图如图1-2所示。由图可知,原型法与生命周期法有很大的不同。原型法可以分为四个阶段:确定用户的基本要求;开发初始原型;实现并使用原型;修改和补充原型。上述第三阶段和第四阶段重复进行,直到获得用户和开发者满意的原型为止。

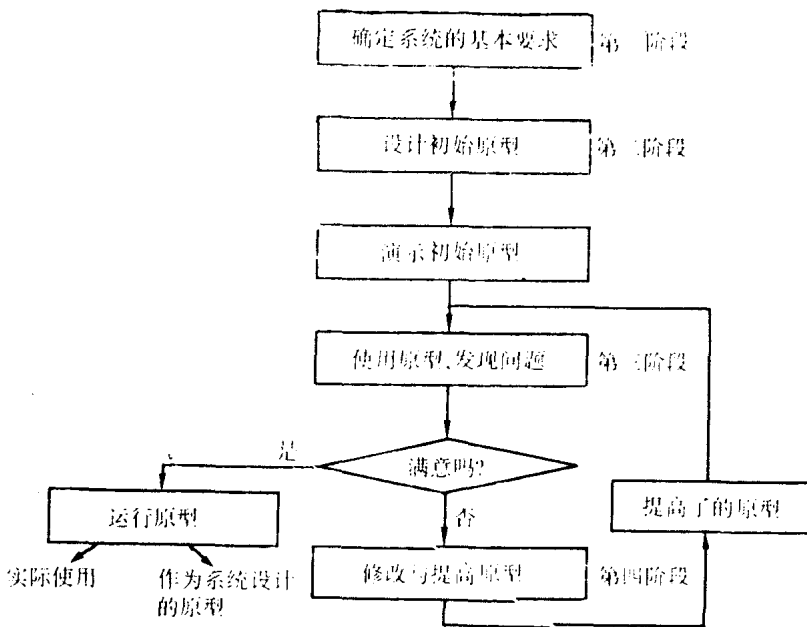


图 1-2 原型法开发过程

原型法的主要特点是开发过程重复进行,不断反馈,总可获得较满意的解。开发过程中,用户和开发人员总在运行一个实际的系统,系统构造灵活,形象直观,便于修改,使得开发周期大为缩短。特别是在构造原型时,若使用第四代语言(4GL),则其效率要比第三代语言高得多。所以,原型法较适用于管理体制和结构不稳定的系统,也适合于开发决策支持系统。原型法的缺点是最初的原型系统的设计比较困难,而且开发过程中缺乏管理和控制手段,也缺乏较完整的文档。因此,大型软件系统难于应用原型法。此时,可以采用

生命周期法为主、原型法为辅的方式进行。如子系统开发时可采用原型法,以缩短系统开发周期。

四、软件开发工具

为了得到软件产品,只有方法学是不够的,还必须有合适的开发工具。

显然,程序设计语言是一种必不可少的工具。此外,还应有其他一些支持工具。

本书的所有软件均采用 dBASE III 和 BASIC 开发。dBASE III 是广泛用于微机上的第四代语言,其数据处理能力,比第三代语言,如 COBOL、PASCAL、PL/1 要强得多,用来开发应用软件的效率要比其他语言高出 10 倍左右。但它的计算能力和作图能力较弱,而 BASIC 在这两方面恰具优势。因此,当前在使用微机时采用这两种语言作为工具是适宜的。

此外,尽管目前国内外在研究和集成化的软件工程支撑环境,即成套的自动化开发工具,以便象在流水线上生产电视机那样,高效率地生产软件产品,但离实用还相差太远。就国内实际情况而言,软件开发仍处于手工生产方式。因此,如能开发出一些通用的有效的手工工具也将是极有价值的。为此,本书第三章开发了若干通用模块,形成了一个软件工具箱,用以支撑各类管理软件的开发,可以大大缩短开发周期,减少开发费用,并能提高软件的质量。它是很有价值的。随着软件工程的发展,将会产生更多的手工工具和自动化工具,以实现软件工程的目标。

§ 1.3 系 统 分 析

经验表明,在能够合适地构造新事物之前,必须作仔细的分析。而系统是事物的汇集,因而彻底分析变得更为重要。

系统分析主要指软件需求分析。通常,在进行需求分析之前,还要进行问题定义和可行性研究。问题定义就是确定软件系统要解决的问题是什么,指出问题的性质、目标和规模,并形成软件计划。可行性研究即探讨软件计划阶段所确定的问题是否有行得通的解决办法。为此,应提出多种可供选择的方案,并从经济上、技术上、社会上等方面进行可行性研究,以确定一个较好的方案。当问题无解时,应建议中止开发。其结果写成可行性研究报告。一般可行性研究的成本占预算工程总成本的 5~10%。若可行性研究报告经评审,确认一个可行方案后,即进入需求分析阶段。

系统分析所依据的是软件工程的基本原理,主要是抽象、信息隐藏、一致性和完整性等基本原理。应对系统进行全面的分析研究,了解现行系统的状况,如系统的规模、内在联系、与外部的关系、组织机构、业务流程、数据流程等,以及系统的运行机制和存在的问题。在此基础上,充分了解用户对新系统的期望、要求和对现行系统的改进意见,从而确定新系统的目标与功能,将现成系统的模型转换成新系统的逻辑模型。这是系统分析的任务。系统分析的目的则在于真正了解用户将要建立一个什么样的系统,从新系统中要得到什么?从而为建立系统作准备。

要指出的是,弄清用户的需求是系统分析的核心。实践证明,许多系统不可靠,起源于不了解用户的要求。调查表明,软件系统中有 15% 的错误起因于错误的要求。

用户要求通常包括功能要求、性能要求、故障处理及恢复要求和限制条件等等。其中功能要求是最主要的要求,务必完整明确。系统分析的结果是系统说明书(需求分析报告)。它是用户和分析人员相互协作,共同劳动的成果。它明确地描述了用户的要求和新系统的逻辑模型,是以后各阶段工作的基础。