

应用软件系统开发

柴跃廷 刘 义 编著
李芳芸 审阅

清华大学出版社

<http://www.tup.tsinghua.edu.cn>



TP317

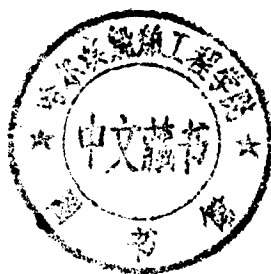
C20

445488

应用软件系统开发

柴跃廷 刘 义 编著

李芳芸 审阅



00445488

清华大学出版社

(京)新登字 158 号

TS168/5/16

内 容 简 介

本书以实际工程需求为背景,系统地介绍了开发各类应用软件系统,特别是大型应用软件系统所必需的基础知识、基本方法与技能。全书分为应用软件开发基础和开发方法两部分。前者包括第1~5章,分别介绍了应用软件的基本概念及软件工程模式;UNIX操作系统的结构与功能、进程的概念及其程序设计技术;编译原理的基本概念和词法、语法分析方法及程序设计技术;信息模型及数据库的概念及其设计方法;网络与分布式系统的基本概念和分析设计方法。后者包括第6~8章,分别介绍了结构化系统分析与设计方法;面向对象的系统分析与设计方法;应用软件系统的开发过程及案例分析。

本书可作为高等院校本科生、研究生应用软件类课程的教材,也可作为从事应用软件开发工作的工程技术人员参考书。

版权所有,翻印必究。

本书封面贴有清华大学出版社防伪标签,无标签者不得销售。

书 名: 应用软件系统开发

作 者: 柴跃廷 刘 义

出版者: 清华大学出版社(北京清华大学校内,邮编 100084)

<http://www.tup.tsinghua.edu.cn>

印刷者: 北京密云胶印厂

发行者: 新华书店总店北京发行所

开 本: 787×1092 1/16 印张: 23 字数: 540 千字

版 次: 1999 年 7 月第 1 版 1999 年 7 月第 1 次印刷

书 号: ISBN 7-302-03472-9/TP·1891

印 数: 0001~4000

定 价: 24.00 元

前 言

计算机技术已广泛地应用到社会生活的各个方面,改变着人们的生产与生活方式,使人类迈入了一个全新的信息时代。信息时代的到来又加剧了社会生产各个领域中的竞争,加快了社会生活的节奏。同时也对计算机及其应用技术在深度和规模等方面提出了更高、更大的要求。我们知道,对于计算机应用系统而言,硬件是躯体,软件是灵魂。这就是说,应用软件系统在计算机应用的各个领域起着举足轻重的作用。

应用软件系统,特别是大规模应用软件系统的开发是一项综合性、智能性的系统工程。它不仅要求开发人员具有扎实的计算机方面的基础知识、基本技能与方法,而且要求开发人员具有相关应用领域的专业知识和一定的组织管理能力。因此,为适应时代的需求,迫切需要培养与造就大量高素质的从事应用软件系统研究与开发的工程技术人员。本书的编写目的也正在于此。

本书是作者在多年从事应用软件系统分析与设计课程教学及工程实践的基础上,对课程讲义进行整理、补充和完善,并结合工程实践的体会编写而成的。全书分为应用软件开发基础和开发方法两部分共8章。应用软件开发基础部分包括第1~5章,分别介绍了应用软件的基本概念及软件工程模式;UNIX操作系统的结构与功能、进程的概念及其程序设计技术;编译原理的基本概念和词法、语法分析方法及程序设计技术;信息模型及数据库的概念及其设计方法;网络与分布式系统的基本概念和分析设计方法。应用软件开发方法及案例分析部分包括第6,7,8章,分别介绍了结构化系统分析与设计方法;面向对象的系统分析与设计方法;应用软件系统的开发过程及案例分析。

书中各章由柴跃廷执笔,刘义对各章进行了修改、补充与完善,李芳芸教授审阅了全书。由于应用软件系统开发涉及的内容非常多,每部分内容都是计算机学科中的一门相对独立的课程。同时,各部分内容本身也在不断发展,日新月异。因此,编写这样一本综合性教材不可能覆盖方方面面,缺点和问题在所难免,望广大读者对本书提出宝贵意见。

作 者

1998年9月

目 录

上篇 应用软件系统开发基础

第 1 章 概述	3
1.1 应用软件的概念	3
1.1.1 软件的概念	3
1.1.2 软件的特征	3
1.1.3 软件的种类	5
1.2 应用软件系统的开发	6
1.2.1 系统开发人员	8
1.2.2 应用软件开发基础	8
1.2.3 系统开发方法	8
1.3 现代软件工程	10
1.3.1 软件工程的概述	10
1.3.2 传统软件工程模式	10
1.3.3 现代软件工程模式	15
第 2 章 UNIX 操作系统及相关程序设计	18
2.1 概 述	18
2.1.1 UNIX 系统的结构与组成	18
2.1.2 UNIX 系统的运行	19
2.1.3 UNIX shell 及其编程	22
2.1.4 UNIX 系统的特点	33
2.2 进程与存储管理	33
2.2.1 UNIX 操作系统中的进程	33
2.2.2 进程管理与调度	40
2.2.3 存储管理	45
2.3 文件及设备管理	46
2.3.1 UNIX 文件	46
2.3.2 UNIX 文件系统结构	46
2.3.3 UNIX 文件系统的存储结构	47
2.3.4 UNIX 文件系统的内部管理与控制	48
2.3.5 UNIX 系统中的设备	54
2.3.6 UNIX 设备管理的特点	55
2.3.7 UNIX 块设备管理	57

2.3.8	UNIX 字符设备管理	58
2.4	系统调用及其程序设计	59
2.4.1	概述	59
2.4.2	进程系统调用及其程序设计技术	59
2.4.3	进程通信及其程序设计技术	68
第3章	编译基础	104
3.1	概述	104
3.1.1	编译程序的作用	104
3.1.2	编译程序的组成与结构	105
3.2	词法分析	106
3.2.1	词法的形式化描述	106
3.2.2	词法分析程序的设计	110
3.2.3	词法分析程序的自动生成	112
3.3	语法分析	121
3.3.1	文法与语言	121
3.3.2	递归下降法	126
3.3.3	算符优先法	129
3.3.4	优先数法	133
3.3.5	LR 分析法	134
3.3.6	语法分析程序的自动生成	139
3.4	语义处理及中间代码	141
3.4.1	概述	141
3.4.2	中间代码的形式	142
3.4.3	几种主要语句的四元式中间代码	144
第4章	信息模型与数据库设计	147
4.1	信息模型的概念	147
4.1.1	问题的提出	147
4.1.2	数据与信息	148
4.1.3	信息模型	149
4.2	建立信息模型的方法	151
4.2.1	ER 方法	151
4.2.2	IDEF _{1X} 方法	156
4.3	从信息模型到数据库的设计	167
4.3.1	基于 ER 模型的数据库设计	167
4.3.2	基于 IDEF _{1X} 模型的数据库设计	174
第5章	网络与分布式系统设计基础	175
5.1	网络拓扑结构与数据交换技术	175
5.1.1	网络拓扑结构	175

5.1.2 数据交换技术	176
5.2 网络体系结构	181
5.2.1 协议及体系结构	181
5.2.2 开放系统互连参考模型(OSI 模型)	181
5.3 局域网	183
5.3.1 局域网的概念	183
5.3.2 局域网中几种常用的介质访问控制方法	185
5.3.3 几种高速网络技术	187
5.4 网络设备与网络互连	189
5.4.1 网络设备	189
5.4.2 网络互连	193
5.5 Internet/Intranet	194
5.5.1 TCP/IP	194
5.5.2 Internet	200
5.5.3 Intranet	208
5.6 网络系统的分析与设计	212
5.6.1 网络系统的分析	212
5.6.2 网络系统的设计	218
5.7 分布式系统设计基础	221
5.7.1 分布式系统的概念	221
5.7.2 分布式系统设计要点	223

下篇 应用软件系统开发方法

第 6 章 结构化系统分析与设计方法	235
6.1 概述	235
6.1.1 基本思想	235
6.1.2 系统分析	236
6.1.3 系统设计	236
6.2 结构化系统分析方法	237
6.2.1 数据流图(DFD)	237
6.2.2 IDEF ₀ 方法	240
6.3 结构化系统设计方法	251
6.3.1 系统软件结构的描述工具——结构图	251
6.3.2 系统软件结构的设计方法	254
第 7 章 面向对象的系统分析与设计方法	268
7.1 基本概念	268
7.1.1 对象	268
7.1.2 类及实例	269

7.1.3	继承性	270
7.1.4	多态性	273
7.1.5	封装性	275
7.2	面向对象的开发方法	276
7.2.1	系统开发及其复杂性	276
7.2.2	系统的模型化及对象化	277
7.2.3	面向对象的开发方法	278
7.3	系统分析	281
7.3.1	概述	281
7.3.2	建立系统需求模型	282
7.3.3	建立系统分析模型	292
7.4	系统设计	302
7.4.1	概述	302
7.4.2	建立系统的设计模型	303
第8章	应用软件系统开发过程	315
8.1	概述	315
8.1.1	应用软件开发流程	315
8.1.2	应用软件系统开发过程	317
8.2	系统分析过程	320
8.2.1	系统分析的目标	320
8.2.2	系统分析的内容与过程	321
8.2.3	系统分析文档	322
8.3	系统设计过程	323
8.3.1	系统设计的目标	323
8.3.2	系统设计的内容与过程	326
8.3.3	系统设计文档	329
8.4	系统实施过程	330
8.4.1	系统实施的内容及过程	330
8.4.2	系统实施文档	332
8.5	案例分析	335
8.5.1	系统需求分析	335
8.5.2	系统结构分析	335
8.5.3	系统配置	341
8.5.4	制定计划及效益分析	352
8.5.5	确定系统自动化程度及人机分工	352
8.5.6	系统信息设计	352
8.5.7	系统软件结构设计	354
8.5.8	模块设计	356
主要参考文献		357

上篇

应用软件开发基础

第1章 概 述

1.1 应用软件的概念

1.1.1 软件的概念

软件是计算机程序加上该程序的各种文档。

软件和计算机程序是不同的概念,程序是使计算机实现所预期的目的(例如解某一算题或控制某一过程)而编排的一系列步骤。或者说,程序是以某种形式的计算机语言(可以是机器指令,也可以是C,C++等高级程序设计语言)表达的解决某种问题的步骤或顺序。计算机程序只是软件的一部分。强调各种文档的重要性,是因为构成软件的文档如同机械产品的设计图纸和规格、使用说明一样,对于机械产品,离开了产品的设计图纸和规格、使用说明,就不可能正确地使用、维修、改造产品。同样的道理,软件离开了各种文档,使用人员就无法正确地使用、维护软件,更谈不上更新、改造软件。软件作为一种特殊的产品,计算机程序及其各种文档是缺一不可的。因此,软件是计算机程序加上该程序的各种文档。

1.1.2 软件的特征

了解软件的特征,有助于深入理解软件的概念,并最终能真正理解软件的开发过程及其特殊性,有助于开发高质量的符合用户需求的应用软件系统。软件不同于硬件,一般具有如下特征:

(1) 软件是一种逻辑性系统元素,不是物理性系统元素,它具有“非实物”属性。在开发硬件时,人们的创造性活动(分析、设计、构造、测试等)最终被转变成有形的物理产品。例如开发一种新型的计算机硬件系统,人们所做的初始草图以及随后的正式设计图纸和原型系统,最后将转化为使用超大规模集成电路的各种印刷电路板和有关电源、机壳等组成的有形物理产品。而软件产品和硬件不同,设计软件产品使用的逻辑量要比硬件产品多10~100倍,为了完成一个复杂的软件系统,需要建立庞大的逻辑体系,其最终的形态表现为计算机程序及其各种文档,溶入其中的人们的创造性工作是无形的,只能通过执行计算机程序完成人们预期的功能,达到预期的性能体现出来。

(2) 软件只能设计或实现出来,不能用传统意义上的制造进行生产。尽管软件开发与硬件开发有某些类似之处,但两者之间有着根本性的区别。首先,根据系统设计生产出来的硬件产品,其质量由于生产过程的变异而不同,但是软件一经设计和实现后,其不同的拷贝的质量是相同的,即涉及不到工艺上的差异。其次,硬件产品由于质量问题可能报废,

而软件产品的质量问题的可以通过将实现过程中的错误予以纠正来解决。也就是说,软件在质量保证方面比硬件具有更大的灵活性。当然,为了提高软件的开发效率,研究人员致力于采用软件工程的思想与方法。借鉴制造业、建筑业的哲理,建立以软件分析、设计、开发、测试工具为主要内容的“软件工厂”,实现软件开发的半自动化或自动化。随着软件工程的不断成熟和软件技术的不断发展,这种思想是可行的,同时也是软件产业发展的必由之路。即使如此,软件开发也不能或者说也不可能完全照搬制造业和建筑业的思想和做法。只有深刻认识软件和硬件生产过程的差异,逐步建立起软件模块或“软件零部件”的国际标准与规范,实现应用软件系统与“软件零部件”开发的分离与融合,才能真正地逐步提高软件开发效率。

(3) 软件不会“磨损”,而硬件却会磨损。对于硬件,在其生命周期的初始阶段存在相当高的故障率,这时出现的故障,常常是由设计或制造过程中出现的失误造成的,当这些研制过程中出现的错误被纠正后,硬件产品趋于定型,因而故障率便下降到一个较低的水平,并在一定时间内相对稳定不变。随着时间的推移,使用中的硬件的故障率开始上升,这往往是由于遭受灰尘的污染以及震动、使用不当、极限温度等许多不良环境的累积效果引起的,简而言之,就是硬件出现了磨损现象。硬件故障率随时间变化的曲线如图 1.1.1 所示。

对于软件而言,在其生命周期的初始阶段也存在有相当高的故障率,出现的故障主要是由于分析、设计阶段不符合需求的部分、开发阶段编程的错误等引起的。当开发过程中的错误被纠正后,其故障率便下降到一定的水平并保持相对稳定,直到该软件废弃不用。理想的软件故障率如图 1.1.2 所示。

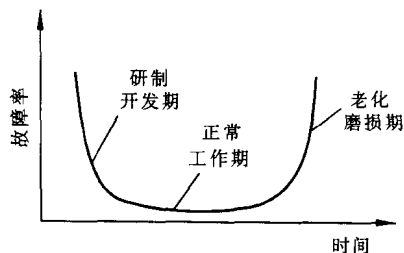


图 1.1.1 硬件故障率曲线

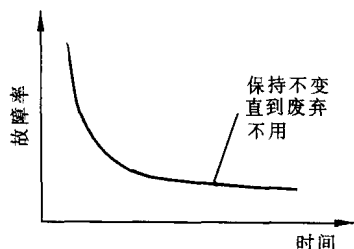


图 1.1.2 理想的软件故障率曲线

实际上,尽管软件不会磨损,但它却会退化。这是因为在软件的生命周期中,由于需求的不不断变化,其本身将会被不断地修改(维护),此时便很可能引入新的故障,导致故障率突然上升,待故障被纠正后,故障率便回落,并趋于后来的相对平稳的故障率。当又有新的需求出现后,软件被修改而可能引入的故障又使故障率突然上升。如此这般循环往复地波动着,从而导致软件故障率的最小值随着时间的推移而缓慢地增加。可以说,由于修改,软件在逐渐地退化。如图 1.1.3 所示。

硬件和软件还存在下述区别:当某一硬件元素磨损后,可以用相同的备用元素更换,从而使硬件系统恢复正常工作。而软件却没有相同的备用元素可言,因为软件出现的每一个故障,要么是由于设计考虑不周造成的,要么是编程错误造成的。由于软件无备用元素可供更换,因而软件维护比硬件维护要复杂得多。

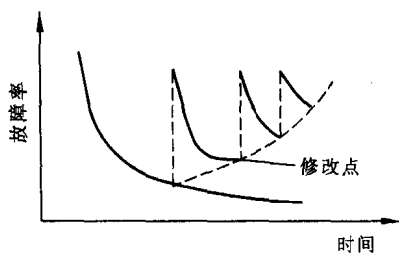
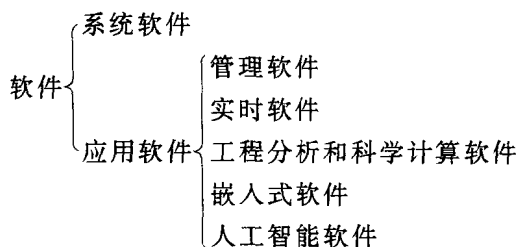


图 1.1.3 实际的软件故障率曲线

1.1.3 软件的种类

按照不同的原则和标准，我们可以将软件划分为不同的种类，但从应用的角度出发，通常将软件划分为系统软件和应用软件两大类，应用软件又可根据应用的不同领域和不同功能划分为若干子类。如下所述：



(1) 系统软件

系统软件是给其它程序提供服务的程序集合（例如各种操作系统、编译程序、文件管理等实用工具）。一般来说，有些系统软件（例如编译程序）处理过程复杂，但其处理的信息比较确定，而且信息结构化程度高；有些系统软件（例如操作系统）处理的信息，在很大程度上是不确定的。系统软件的主要特征是：与计算机硬件系统有很强的交互性，并且要对共享资源进行调度管理，因而系统软件具有复杂的过程管理程序以解决并发性操作处理中存在的协调问题。而且系统软件中的数据结构复杂，外部接口多样化，以便于许多用户反复使用。

(2) 管理软件

管理信息处理是最大的软件应用领域，管理信息系统往往集成各种分离的应用软件，例如将生产管理、计划管理、工资管理、财务管理、物料管理等分离的应用软件集成起来以支持企业的各项管理工作。管理信息系统应用一个或多个大型数据库管理系统来存放有关管理信息，并对数据库中存放的数据进行适当的处理，以辅助经营管理和决策。管理信息系统除了完成传统的数据处理任务外，还应提供交互式处理功能，例如对客户订单、销售订单等的处理必须以交互式方式进行。

(3) 实时软件

对现实世界中随时发生的事件进行监测、分析和控制的软件，称为实时软件。实时软件一般有助于收集数据的基本模块、分析模块和监控模块。收集数据的模块从外部环境中

收集信息,并将其结构化;分析模块把收集来的信息变换成相应的应用需要的信息;监控模块根据分析模块提供的信息对外部环境做出正确的响应,并协调收集数据模块,保证系统的实时响应(一般要求的响应时间范围是 1ms 到 1min 之间)。

(4) 工程分析和科学计算软件

工程分析和科学计算软件是以对数据进行按部就班的处理为特征的。它的应用范围非常广泛,从天文学研究到火山研究,从自动应力分析到空间运行轨道动力学分析,从分子生物学研究到自动生产过程控制等。新的工程分析与科学计算软件已扩展了传统的数学运算的概念和范围,例如计算机辅助设计、系统仿真以及其它的交互式应用软件。

(5) 嵌入式软件

现在,智能产品几乎出现在所有的家用市场和工业市场上。智能产品中所谓的智能主要是因为产品中嵌入了具有一定程度的智能功能的软件。一般来说,嵌入式软件仅完成一些非常有限但又比较奥妙的功能,例如微波炉上的操作键的控制。或者提供一些重要的功能和控制能力,例如,汽车中使用的各种数字显示功能、油门控制以及制动系统等。

(6) 人工智能软件

人工智能软件利用非数学算法解决复杂问题,这些问题往往不是单纯用计算和直觉分析所能解决的。当前,人工智能领域最活跃的是专家系统,也称为基于知识的问题求解系统。人工智能软件还有一些范例,例如模式识别、理论证明等。

软件的应用领域还很多,这里就不一一列举了。总之,凡是应用到某一领域,利用计算机技术解决某个(或某些)实际应用或辅助人们更好地从事工作的软件,统称为应用软件。随着计算机技术的发展、信息时代的到来,对应用软件的需求将日益增多。因此,研究应用软件的分析和设计技术,提高应用软件的开发效率和质量具有重要的现实意义,也是本书的目的所在。

1.2 应用软件系统的开发

讨论应用软件系统的开发问题的最终目标是以最短的时间、最低的成本、最高的质量开发出满足用户需求的应用系统,即开发出一个“好”系统。那么,什么是好的系统呢?一般而言,一个好的系统须从两方面来衡量。一是从系统外部来看,一个好的系统对系统的所有用户而言,应该能够快速给出用户所期望的正确结果,而且是可靠的,并易于用户有效地掌握和使用。一是从系统内部来看,一个好的系统,对系统的所有开发者都应易于理解、修改、测试,具有一定程度的可重用部件,易于适应变化的需求,同时与其它系统具有很好的兼容性,易于移植。

一个好的系统对于不同的应用可能具有不同的衡量标准。例如,对于实时软件可能要求系统具有很好的实时性;对于管理软件可能要求系统具有很好的人机界面。但不论是什么类型的应用系统,决定一个系统好坏的关键因素在于开发初期对系统需求的初始描述及系统的体系结构。系统需求的初始描述及系统体系结构的好坏实质上体现了系统的有序程度,有序程度越高,系统的初始描述及体系结构越好,反之,则越坏。由热力学第二定律可知,系统的有序程度是用系统的热量来衡量的,系统热量越大,系统越无序,反之,越

有序。类似地,我们用软件能量来描述应用软件系统的有序程度。开发经验表明:在对软件系统进行修改与维护时,软件能量增加与软件系统的初始能量呈比例。即

$$\Delta E \sim E$$

其中, ΔE 表示软件能量的增加, E 表示软件系统的能量,图1.2.1显示了这一规律的作用。

由图1.2.1可知,系统的生命周期取决于系统的初始能量,即取决于系统的初始描述与体系结构。系统的初始描述与体系结构越好,系统越有序,能量越低,系统的生命周期越长。在对系统开发与维护过程中,系统的能量逐渐增加,当增加到某一极限量,意味着系统的维护费用已高到一定程度,已不值得再对系统进行维护,只有丢弃系统或对系统进行重新设计以降低系统的能量,这意味着系统生命周期的结束。

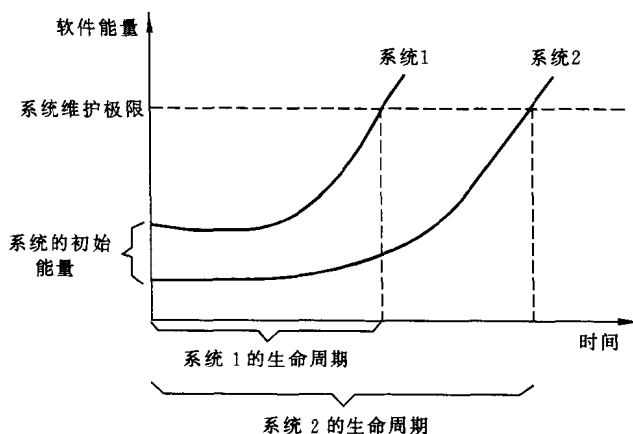


图 1.2.1 软件初始能量对系统生命周期的影响

因此,我们分析与设计应用系统的目标是致力于使系统具有较低的初始能量,较长的生命周期。要做到这一点,不是一件容易的事情,因为应用软件系统的开发是一个复杂的过程,这个过程不仅包含了系统需求分析、设计、实现、测试、维护等子过程,而且要求这些子过程的有机集成。同时,对每个子过程而言,都需要开发人员、软件基础及开发方法与工具的密切配合,如图1.2.2所示。

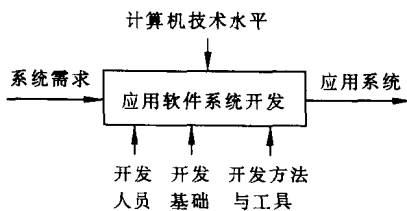


图 1.2.2 应用软件开发

应用软件的开发过程是在现行的计算机硬件、软件技术水平约束下,在有关系统软件(例如操作系统、编译程序、数据库管理系统等)的基础上,由开发人员采用一定的开发方

法和开发工具将系统的需求转换为应用系统的过程。

1.2.1 系统开发人员

系统开发人员是指完成系统开发任务的项目组成员,由系统分析与设计人员(包括程序员)和用户组成。系统分析与设计人员是完成系统分析与设计任务的主要承担者和任务协调者,用户是指系统最终用户或其代表。

1.2.2 应用软件开发基础

应用软件系统是运行在计算机硬件、系统软件和网络基础之上的。因此系统软件及网络就构成了应用软件分析、设计和实现的基础。因为应用软件系统的实现更多地依赖于系统软件和网络。

应用软件开发基础主要包括操作系统、编译程序、数据库管理系统及其设计技术、网络与分布式系统设计基础等。

对于操作系统,主要有两大类,一类是基于桌面处理的单用户操作系统,例如 DOS, WINDOWS 等,另一类是基于多进程的多用户操作系统,例如 UNIX, WINDOWS NT 等。对于大型复杂的应用软件系统,多用户操作系统是必不可少的。在多进程多用户操作系统环境下,可以完成比较复杂的系统处理,为系统前台提供更好、更多的透明服务。因此,利用多进程多用户操作系统提供的诸如进程管理及应用程序编程接口设计应用软件系统中的复杂功能是应用软件系统开发中的重要方面。

对于编译程序,在应用软件系统的人机交互界面和内部功能协调方面,具有重要的应用价值。例如,利用编译程序中提供的词法、语法分析方法,可以设计一些面向最终用户或内部模块之间的操作与调用命令,实现应用功能的词法、语法分析等。

对于数据库管理系统,不论是管理软件,还是实时软件,都需要有数据库管理系统的支持,完成一些临时或永久信息的保存与检索功能。在应用软件开发中,掌握数据库的设计技术是必不可少的一环。

对于网络与分布式系统设计基础,由于大型复杂的应用软件系统大多是分布式应用系统。计算机网络是其赖以存在的基础,掌握网络的基本概念及网络系统的结构、功能、必要的编程接口是开发应用软件必备的基础。利用分布式系统的基本概念与设计方法,合理分析与设计应用软件系统的体系结构、功能分布、信息分布是设计高质量、高性能应用系统的重要前提条件。

综上所述,操作系统、编译程序、数据库系统设计技术、网络与分布式系统设计技术是开发应用软件系统的重要基础,也是系统开发人员必备的基础。因此,在本书的第一部分将详述有关内容。

1.2.3 系统开发方法

开发高质量的应用软件系统可以采用多种方法,不同的方法具有不同的特点。目前,用于系统开发的方法可以分为两大类,一类是结构化方法,一类是面向对象方法。

1. 结构化开发方法

结构化开发方法是一种传统的方法,已为广大的系统开发人员所熟悉,其基本要点是:

(1) 采用自顶向下、逐步求精的思想,将系统功能按层次进行分解。对系统开发人员,可以很方便地设计与实现系统。而对于用户,不但对系统有一个系统的概念和认识,而且可以了解系统某个具体的组成部分及其功能细节,能够及早地看到结果,及时地提出意见。

(2) 吸收用户参与系统分析与设计。在系统分析阶段,强调系统能够“做什么”,在系统设计阶段,强调系统“如何做”,界面清晰,易于被开发人员及用户理解。

(3) 采用图示工具附加语义说明的方法阐述系统的结构与功能,便于系统开发人员和用户之间的相互交流,对系统有一致的理解。

适合于结构化分析与设计技术的主要方法有:

- 数据流程图法(DFD)
- 结构化分析/结构化设计方法(SA/SD)
- IDEF 方法
- Jackson 方法

结构化开发方法的核心在于将系统中的功能与数据相分离,这种方法在系统功能和数据结构由于系统需求变化而发生变化时会导致一系列问题,而系统需求变化是绝对的,系统功能和数据结构对于系统需求变化是最敏感的。该方法的最大的问题是系统难于修改和维护,这是因为系统中所有的功能必须知道数据是如何组织的,也就是数据结构。如果改变数据结构,必须相应地改变与之相关的功能。因此常常会导致系统结构不稳定,系统中一个小的变化可能会导致严重的后果。

2. 面向对象开发方法

面向对象开发方法是继结构化开发方法之后,着重解决系统需求变化和维护困难等问题而发展起来的一种全新的方法,其基本要点是:

(1) 在系统分析阶段,用问题空间中的实体(或对象)及其相互关系建立系统逻辑模型,以一种更加自然的方式描述系统的需求及体系结构。由于问题空间中的对象是相对稳定的,即使有变化,通常只影响系统的局部或将系统需求的变化局部化,从而使系统具有一个相对稳定的结构。

(2) 系统设计与实现是对分析阶段得到的逻辑模型的细化或实例化。在逻辑模型中的对象,最终以代码的形式出现在设计与实现模型中,使得分析与设计具有无缝连接,系统易于测试与维护。

(3) 系统开发过程是系统需求、分析、设计、实现、测试等一系列模型的细化过程,可以有效地解决系统复杂性问题。同时,构成系统的基本单元——对象有利于被设计成具有标准界面的软件构件,使之具有一定程度的可重组性,为软件开发的规模化、工业化奠定基础。