

应用软件开发技术

刘俊

黑龙江人民出版社

应用软件开发技术

刘 俊 编著

黑龙江人民出版社

图书在版编目(CIP)数据

应用软件开发技术/刘俊 编著

哈尔滨:黑龙江人民出版社.2006.6

ISBN 7-207-06911-2

1、应…II、刘…III、软件开发—技术—中国 IV1217.3

中国版本图书馆CIP数据核字(2006)第017786号

责任编辑:安晓峰 装帧设计:午夜

应用软件开发技术 刘俊编著

出版:黑龙江人民出版社

通讯地址:哈尔滨市南岗区宣庆小区 1号楼

邮编:150008 印刷:竺桥印务

开本:850×1168 1/32 印张 10

字数 210千字 印数 1000

版次 2006年6月第一版 2006年6月第一次

书号:ISBN-7-207-06911-1/2.923 定价:28.00元

前 言

从1969年提出软件工程概念以来,历经30多年的飞速发展,软件工程学已经逐渐成熟,现已成为计算机科学与技术领域中一门重要的学科。应用软件开发技术的目标是以解决软件生产的质量与效率问题为宗旨,研究一套科学的工程技术方法,以及与此相应的方便的软件工具系统,用来指导和帮助软件的开发与研究,在软件的开发与研究起到重要的技术保障和促进作用。

本书分为8章,第1章 应用软件开发过程概述,第2章 面向对象技术,第3章 统一建模语言UML,第4章 应用软件开发需求分析,第5章 应用软件开发原则与方法,第6章 面向对象程序设计,第7章 应用软件开发的质量保证,第8章 应用软件开发工具与环境。

本书系统地介绍了应用软件开发技术的主要内容,可作为从事软件开发与应用的工程技术人员的参考书,也可作为高等院校相关专业的教材或参考书。

在本书编著过程中,吸收了许多专家、学者的研究成果,借鉴了相关著作的特长,同时还得到了本教研部多位同事的大力支持和帮助,在此一并表示真挚的感谢。

由于作者水平有限,书中不足之处在所难免,敬请批评指正。

刘 俊

2005.11 于南京

目 录

第 1 章 应用软件开发过程概述	1
1.1 应用软件开发过程	1
1.1.1 开发前的准备工作	1
1.1.2 应用软件开发过程	3
1.1.3 软件开发后的工作	8
1.2 应用软件开发模型	9
1.2.1 瀑布模型	9
1.2.2 演化模型	10
1.2.3 螺旋模型	12
1.3 应用软件开发技术基础	14
1.3.1 计算机理论基础	14
1.3.2 计算机应用基础	28
1.3.3 数据库技术	32
1.3.4 计算机网络技术	37
1.3.5 多媒体技术	47
1.4 应用软件开发技术的发展	52
第 2 章 面向对象技术	55
2.1 面向对象的基本概念和特征	56
2.1.1 对象与类	56
2.1.2 基本特征	62
2.2 面向对象的应用软件开发过程	68
2.2.1 RUP 的二维开发模型	68
2.2.2 RUP 的特征	73

第3章 统一建模语言 UML	78
3.1 UML 语言概述	78
3.1.1 UML 语言的由来	78
3.1.2 标准建模语言 UML 的主要内容	80
3.1.3 标准建模语言 UML 的主要特点	82
3.2 UML 的静态建模机制	83
3.2.1 用例图	83
3.2.2 类图、对象图和包	87
3.2.3 构件图和配置图	96
3.3 UML 的动态建模机制	98
3.3.1 消息	98
3.3.2 状态图	99
3.3.3 顺序图	100
3.3.4 协作图	102
3.3.5 活动图	103
3.3.6 四种图的运用	106
3.4 软件过程参考模型 RUP	107
3.4.1 Rational Unified Process 简介	107
3.4.2 二维的软件开发模型	107
3.4.3 静态结构:方法描述	108
3.4.4 动态结构:迭代式开发	110
3.4.5 使用用例驱动	111
3.4.6 以体系结构为中心	112
3.4.7 RUP 的优点	112
3.5 软件开发工具 Rational Rose	113
3.5.1 可视化	114
3.5.2 建模定义	116
3.5.3 用 Rational Rose 实现建模	117

3.5.4 Rational Rose 系列产品	119
第4章 应用软​​件需求分析	120
4.1 需求分析的任务和步骤	120
4.1.1 需求分析的任务	121
4.1.2 需求分析的步骤	123
4.1.3 需求分析的原则	127
4.2 需求分析的方法	128
4.2.1 面向数据流的需求分析方法	130
4.2.2 数据流图	130
4.2.3 数据字典	138
4.3 图形工具	144
4.3.1 层次方框图	144
4.3.2 Warnier 图	145
4.3.3 IPO 图	145
4.4 需求规格说明与评审	147
4.4.1 需求规格说明的主要内容	147
4.4.2 需求分析的评审	148
4.4.3 需求分析评审的方法	150
第5章 应用软​​件设计原则与方法	151
5.1 应用软​​件设计的任务和步骤	151
5.1.1 设计的任务	151
5.1.2 设计的步骤	151
5.2 应用软​​件设计的原则	153
5.2.1 模块化	153
5.2.2 抽象与逐步求精	154
5.2.3 信息隐蔽和局部化	155
5.2.4 模块独立性	155
5.3 应用软​​件设计方法概述	156

5.3.1 面向数据结构的设计方法	156
5.3.2 面向数据流的设计方法	158
5.3.3 面向对象的设计方法	162
5.4 应用软件设计常用工具	185
第6章 面向对象程序设计	188
6.1 程序设计语言	188
6.1.1 程序设计语言分类	188
6.1.2 程序设计语言的特点	191
6.1.3 程序设计语言的选择	193
6.2 程序设计的基本原则	195
6.2.1 编程风格	195
6.2.2 程序效率	200
6.2.3 程序安全	202
6.3 面向对象的程序设计	204
6.3.1 对象的定义	204
6.3.2 建立软件系统的动态模型	206
6.3.3 建立软件系统的静态模型	206
6.3.4 实现	206
6.3.5 编程的定义	206
6.3.6 从设计到 Delphi 代码	208
6.3.7 从设计到持久代码	212
6.3.8 编程技巧	215
第7章 应用软件的质量保证	221
7.1 应用软件质量概述	221
7.1.1 定义	221
7.1.2 应用软件质量的标准	222
7.1.3 影响应用软件质量的主要因素	223
7.1.4 保证应用软件质量的措施、活动	225

7.2 应用软件评审	228
7.2.1 应用软件评审的原则	229
7.2.2 应用软件设计质量的评审	229
7.2.3 应用软件程序质量的评审	230
7.3 应用软件测试	233
7.3.1 测试的目标、原则和策略	234
7.3.2 测试计划	236
7.3.3 结构化软件测试	238
7.3.4 面向对象软件测试	250
7.3.5 测试分析报告	264
7.4 应用软件维护	265
7.4.1 应用软件维护的分类、特点	265
7.4.2 应用软件的维护技术	269
7.4.3 应用软件的维护步骤、维护组织	271
7.4.4 应用软件的可维护性	276
7.4.5 应用软件的维护副作用	278
第8章 应用软件开发工具与环境	280
8.1 开发工具的特性、功能和分类	280
8.1.1 开发工具概述	280
8.1.2 开发工具的功能和分类	281
8.1.3 开发工具的选择	283
8.1.4 常用开发软件比较	288
8.2 软件开发环境	295
8.2.1 软件开发环境概述	295
8.2.2 软件开发环境分类	295
8.2.3 计算机辅助软件工程	298
8.2.4 构件技术的概念及发展	299

第1章 应用软件开发过程概述

随着需求的日益复杂,应用软件的规模也越来越大,因此应用软件开发人员应确立软件开发的工程思想。

1.1 应用软件开发过程

应用软件的开发和所有工程产品的生产一样,要经过分析、设计、制造、检测和运行维护等几个阶段。通常,把软件从开始研制到最终软件被废弃不用这段时期称为软件生存周期。我国国家标准《软件开发规范》从软件生存周期的角度将应用软件开发分为以下八个阶段:可行性研究与分析、需求分析、总体设计、详细设计、实现、集成测试、确认测试、使用和维护。

本节只简要介绍这八个阶段在整个应用软件开发过程中所扮演的角色。

1.1.1 开发前的准备工作

应用软件开发前的准备工作是指可行性研究与分析、需求分析这两个阶段。可行性分析和研究的步骤见图1-1。

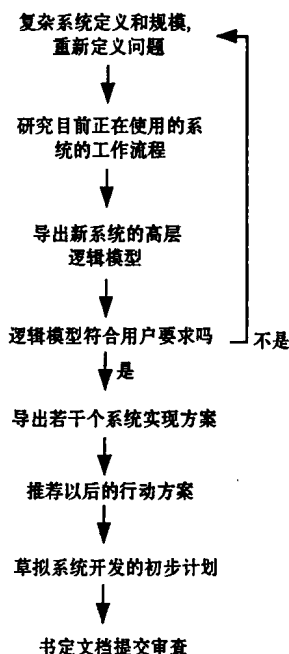


图1-1 可行性研究和研究的步骤

如果用户想使现有的手工工作流程计算机化或对现有的系统做出改进,开发应用软件的需求便出现了。用户根据其工作中的情况提出希望要解决的问题。此时,系统分析人员会同用户对待解决的问题进行分析和讨论,其目的是对待解决的问题做出清晰的定义。在问题定义之后,需要进行可行性研究与分析。可行性研究的目的在于:确定在问题定义阶段所描述的系统目标和规模是否现实可行,是否能够解决和是否值得去解决。通常从应用软件开发的技术可行性、经济可行性、操作可行性以及法律可行性这四个方来考察待开发系统是否可行。

可行性分析和研究需要用户、系统分析人员、使用部门的负责人和专家以及计算机硬件和软件方面的专家的参与。

如果待开发系统的可行性分析和研究的结论是可行的,接下来要进行的便是待开发系统的需求分析。需求分析是整个软件开发的基础性工作,需求分析阶段生成的文档—需求规格说明书,是以后各阶段工作的依据,所以完善的需求分析对软件开发的成败是极为重要的。

需求分析是在可行性研究和分析的基础上进行的,需求分析的基本任务是要准确的回答“系统要做什么?”的问题。需求分析的主要任务有以下三个方面:

(1)正确的确定问题,充分的理解和表达用户的需求。即详细定义待开发软件的功能、性能、外部接口、有关属性、设计限制、数据库需求、确定硬件和软件支持环境以及辅助软件。

(2)使用自顶向下逐层分解的结构分析的方法,把整个软件系统分解为若干个子系统,并定义各子系统的功能以及它们之间的接口。

(3)对上述两项工作进行描述,以形成“需求规格说明书”。

需求分析的步骤见图 1—2。其由系统分析人员、用户、使用部门的负责人等协同完成。这个阶段生成的文档“需求规格说明书”。

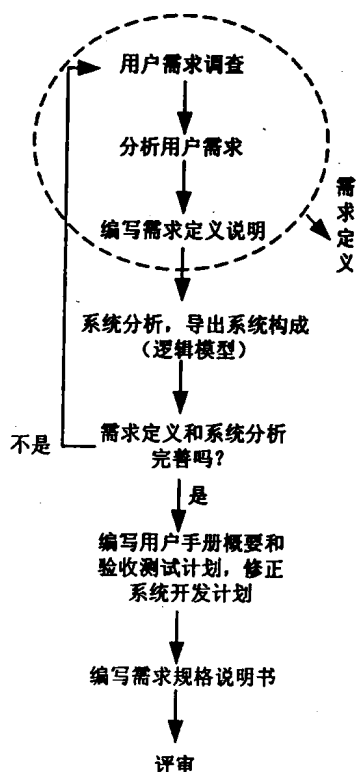


图 1-2 需求分析的步骤

1.1.2 应用软件开发过程

在可行性分析和研究、需求分析完成之后,便进入软件开发的具体设计阶段。其包括软件的总体设计、详细设计、编码和测试。

总体设计又称结构设计,其基本任务是:将系统划分模块、决定每个模块的功能、决定模块的调用关系、决定模块之间的接口。在总体设计阶段,设计人员应遵循以下软件工程学原则:

1. 模块化

现在的软件系统都比较复杂。人们在解决复杂问题时常使用

“抽象”、“分解”的方法,即把一个复杂的问题分解为一个个相对简单的问题。模块化就是对于一个复杂的软件系统进行分解。所谓模块化是指:按一定原则把程序划分为若干个模块,使每一个模块完成一个子功能。

2. 抽象

抽象是对事物的共同的本质的特性进行抽取和概括,忽略其次要因素。抽象是设计人员必须具备的能力,它是模块化的前提和基础。

3. 信息隐藏

在设计和确定模块时,使得一个模块内包含的信息(数据和过程)对于不需要这些信息的模块来说是不能访问的。设计人员在划分模块时,要问自己这样一个问题:应该怎样划分模块才能得到高质量的软件系统? 高质量的软件系统是指该软件系统可维护性好、可靠性高、可理解性强和效率高。这四个指标均与模块划分时的信息隐藏程度密切相关。

4. 信息局部化

将一些关系密切的软件元素(数据和所有有关的过程),物理的放得彼此靠近。所谓“关系密切”是指这些软件元素是共同解决某问题或实现某功能所需的。

5. 一致性

软件系统的文档、程序使用一致的符号;对操作有一致的控制结构和调用序列;在任一抽象层次中逻辑相关的对象的表达是相同的。

6. 完整性

一个抽象成分的所有本质的东西都明显的定义和描述了,没有本质的东西被遗漏掉。

7. 确定性

划分模块时应保证所有模块能被容易的测试,对这些模块的

定义应是明确的,不存在二义性。

总体设计一般由系统分析员和高级程序员完成。

总体设计完成之后,软件开发就进入详细设计阶段。详细设计即模块设计,其以总体设计为基础和依据。详细设计阶段的任务就是为每个模块进行详细的算法过程设计、内部数据结构设计、程序逻辑结构设计。

详细设计一般由高级程序员和程序员担任。

编码是在详细设计的基础上进行的。其主要任务是根据详细设计阶段给出的程序逻辑结构的描述,选择某种语言,按照编程规范,编写出高质量的、具有一致性、可移植性、可维护性和高效率的程序代码。

尽管在现在的软件开发当中编码所占用的时间和人力都呈下降的趋势,但编码在整个软件的开发当中仍然是重要的。在编码中需要注意两个问题:

(1) 程序语言的选择

程序语言的选择应考虑以下的因素:

◆ 软件的应用领域。这是选择语言的关键因素,各种语言适用于不同的应用领域。

◆ 系统用户的要求。用户往往希望软件能够使用他们熟悉的语言来编码。

◆ 可以使用的编译程序。运行目标系统的环境可以提供的编译程序限制了对语言的选择。

◆ 程序员的经验和知识。应该尽量选择程序员熟悉的语言来编码。

◆ 软件的可移植性要求。由于现在硬件和软件的发展速度都很快,一个应用软件可能跨越硬件和软件的几个发展周期,因此用户往往希望应用软件能在不同的平台上运行,这就要求选择可移植性强的编码语言。

(2) 程序设计风格

好的程序设计风格应能达到以下要求:

◆ 易于测试和调试。据统计,程序设计风格优良和程序设计风格恶劣的具有同样功能的两段代码,它们在测试和调试上所花费的时间之比为 1 : 28。

◆ 易于维护。

◆ 易于修改。

◆ 设计简单。为使程序易于理解、易于调试、维护和修改,最好的办法是使程序设计简洁,坚决摒弃为了炫耀技巧而使程序设计复杂化的任何想法和做法。

◆ 高效。据统计,一段典型的程序代码 50% 的执行时间被 3% 的语句所占用,所以这部分代码的效率很重要。

◆ 文档完备。这里的文档即包括模块设计卷宗,也包括程序内部的文档。程序内部的文档指代码注释和程序风格。程序代码除了实现模块的功能这个角色以外,它本事也相当于一个文档,要提供给有关的开发人员、测试人员、维护人员以及评审人员阅读。因此,程序代码的可读性相当重要。这就要求程序代码的注释清楚完备,程序风格通俗、简洁、一致。

编码主要由高级程序员和程序员来承担。

任何产品在生产出来以后均需要测试,以确保其质量符合市场和用户的要求。软件产品也不例外,在编码这个阶段结束以后,软件开发便进入软件测试阶段。

随着软件结构日益复杂,软件规模日益庞大。分析、设计和编码人员在软件开发的各个阶段当中很难避免错误。因此,无论怎样强调软件测试的重要性都不为过。据统计,软件开发的总成本中有 30%~50% 用在测试工作上,再加上运行维护阶段的测试工作,估计在软件开发的时间和成本方面均有 50% 的时间用在测试方面。

软件测试的经典定义是 G. J. Myers 提出的,他认为“软件测试是为了发现错误而执行程序的过程”。我们可以根据 G. J. Myers 的定义认为“成功的测试是发现了至今尚未发现的错误的测试”。

在软件测试的过程当中应遵循以下原则:

(1)开发队伍和测试队伍应分别建立。如果开发人员承担测试工作,他们在主管上或潜意识里都不太愿意找出自己工作成果中的错误。而且,不同的人有不同的思维、经验、知识背景和工作方式。如果让另外的人员来担任测试工作,能够更容易的发现存在的错误。

(2)设计测试用例时,要给出测试的预期结果。另外,在用例中除了包括对有效的和预期的输入条件进行测试以外,还应包括对无效的和非期望的输入条件进行测试。

(3)只测试程序是否完成应做的事是不够的,还应测试程序是否做了不该做的事。

(4)在对程序进行修改之后,要进行回归测试。

对程序的任何修改都可能引入新的错误,所以必须进行回归测试。所谓回归测试是指在软件修改之后再次运行以前为查找错误曾用过的测试用例。

(5)对发现错误较多的程序段,应进行更深入全面的测试。

对测试结果进行统计表明:一段程序中发现的错误数越多,则其中存在错误的概率也就越大。

在以前人们的思维当中,软件测试集中于程序代码,在以前软件结构简单,软件规模较小的时候,这种想法无可厚非。但对于现在结构复杂、规模庞大的软件而言,单纯对程序代码测试是远远不够的,测试人员必须对软件开发的各个阶段进行跟踪测试,以便及时发现和更正错误。软件测试的主要对象和依据是各阶段生成的文档,软件测试阶段生成的文档为“软件测试报告”。

1.1.3 软件开发后的工作

这是软件生存周期的最后一个阶段。现在的软件开发,无论是使用单位还是开发单位,均投入了大量的人力、物力。并且,现在的软件存在于一个计算机硬件和软件均更新迅速的时代,软件可能需要适应不同的平台。因此,软件的维护就显得尤为重要。否则,将给软件使用单位带来巨大的经济损失,而软件开发单位也会失去市场和信誉。

美国的 Barry W. Boehm 把软件维护定义为“对现有软件进行修改而同时保留其主要功能不变的过程”。

这里只介绍软件维护中最重要的一类维护,即完善性维护。

所谓完善性维护是指为了提高软件性能和软件可维护性而对软件修改的过程。据 Belady. Lehman 对于软件维护的进化动力学研究指出:

(1) 软件连续不断的变化以反映进化要求

一个正在使用的大型软件要经历连续不断的变化,否则它的用处会日益减少。

(2) 复杂性增加要求进行进化性的维护

因为一个大型软件被不断的修改,所以它的复杂性也相应增加,除非进行工作来维护它。

(3) 软件进化的基本规律

由于大型软件进化动力学的存在,这种进化引起对整体项目的系统属性进行周期性的自我调整。

(4) 软件版本的进化

为了使软件可靠的、有计划的进化,一个经历变化的大型软件,必须周期性按安全的新版本所要求的时间间隔,不断发布最新修改版本。

由 Belady. Lehman 的研究可知,软件的进化存在客观的规律,其使得软件的功能和性能不断完善和扩充。软件在其使用过