

软件工程研究院



刘寅斌
飞思科技产品研发中心

编著
监制

系统

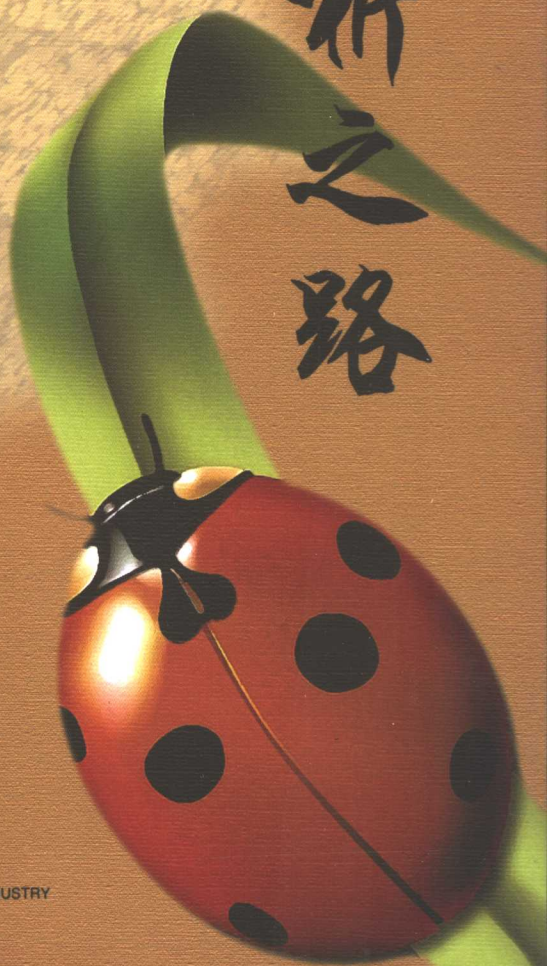
分析之路

THE ROAD OF SYSTEMS ANALYSIS



电子工业出版社

PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
<http://www.phei.com.cn>



软件工程院

系统分析之路

THE ROAD OF
SYSTEMS ANALYSIS

刘寅琥

编著

飞思科技产品研发中心

监制

电子工业出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

系统分析设计工作是软件开发组织开发优秀软件系统的重要的工作,需要拥有很强的理论基础和实际工作经验的人来做。本书结合笔者的工作实践和多年的研究,讲解系统分析设计的思想、过程、相关的理论和方法,以及操作方式。所有的理论和方法都进行了细致的解释,力求帮助读者正确理解,在头脑中建立良好的知识体系结构,而不仅限于知道理论和方法。

本书适合于有编程经验的软件工程师、系统分析理论的研究者和对系统分析感兴趣的人员阅读,也适合于资深系统分析员阅读和参考。同时,本书还适合作为系统分析相关课程的教材或者参考读物。

未经许可,不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有,侵权必究。

图书在版编目(CIP)数据

系统分析之路 / 刘寅琥编著. —北京:电子工业出版社, 2005.5

(软件工程研究院)

ISBN 7-121-01137-9

I.系... II.刘... III.软件工程—系统分析 IV.TP311.5

中国版本图书馆 CIP 数据核字(2005)第 036407 号

责任编辑:赵红梅

印 刷:北京智力达印刷有限公司

出版发行:电子工业出版社

北京海淀区万寿路 173 信箱 邮编:100036

经 销:各地新华书店

开 本:787×1092 1/16 印张:33 字数:844.8 千字

印 次:2005 年 5 月第 1 次印刷

印 数:6 000 册 定价:49.00 元

凡购买电子工业出版社的图书,如有缺损问题,请向购买书店调换。若书店售缺,请与本社发行部联系。
联系电话:010-68279077。质量投诉请发邮件至 zltz@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

前言

欢迎进入系统分析员的世界！这是一个软件技术、系统思想相互支撑的软件开发的世界。好的系统分析员会把一个由破碎的思绪、闪现的灵光、繁琐的需求和缥缈的想像组成的虚幻的系统慢慢地转化为一份符合预算和各方利益的软件实现的方案。而软件开发人员，则通过协调努力完成这份方案。最终，一个能够产生价值的软件系统从抽象的计算世界走了出来，为用户证明着信息的价值和威力。

随着组件技术和新的软件开发工具的投入使用，软件开发人员的开发能力得到了显著提高，同时软件应用领域也迅速地扩展，许多原来认为软件不能完成的工作，现在却天经地义地交给软件系统了。早期的软件只是数值运算而已，后来出现了大规模的数据处理和联机处理业务。等到字符串取代数值成为运算主要的处理对象的时候，计算机也就脱离了科学计算的束缚，进入了社会经济的其他领域。面对那些没有成熟的计算公式和良好算法的领域，甚至是没有良好的问题形式化描述的领域，系统分析设计工作起到了重要的作用。越是复杂的系统越是如此，系统分析员为信息世界制定着运转的规则。显然，正是系统分析员在需求和实现的机制之间架起了一座桥梁，使得信息系统可以很好地完成现实世界需要的功能。好的系统分析员既了解需求那一端，也就是软件应用领域的情况，也了解具体的编程实现技术那一端，也就是实现技术的情况，不仅如此，他还应该具有实现这种映射和变换的能力。

成为系统分析员需要系统分析理论的学习和实际设计经验的积累。在实际项目的设计过程中，包含了许多技术以外的因素，这也是系统分析设计工作重视经验的原因。但是，掌握扎实的理论可以帮助系统分析员面对飞快发展的技术潮流而有所应对，能够迅速地使自己适应新的生产技术环境。在理论学习过程中，如何使用系统分析思想和理论是学习的核心，可以把这样的实际理论和原则叫做“应用理论”。本书介绍的内容就是这种应用理论，也就是结合设计过程中的经验介绍理论的应用和应用时的理论，既包括系统分析设计的思想和理论，又阐述了这些理论和实际的分析设计如何结合，特别针对一些实际的设计过程中的典型问题，给出了经过验证的成熟的解决方法。

本书的内容基本涵盖了系统分析员需要掌握的知识和技能，适合软件程序员、软件工程师、高级软件工程师、系统分析员和项目经理阅读。此外，对于软件开发感兴趣的人员也可以作为参考。依据读者自身特点的不同，可以选择以系统分析设计方法、软件架构、系统分析设计理论、需求捕获方法或者是以软件开发过程管理为主，进行阅读和学习。但是也应该了解其他部分的内容，因为真正好的系统分析员需要了解各个方面的知识。

本书是基于笔者讲述的“面向对象系统分析技术与 CMM”这门课程的内容和体会撰写的，同时结合了笔者的研究成果和其他企业培训的经验。因此，本书可以作为教材，从头至尾地学习。另外，各个章节也相对自成体系，完整地讲述一个问题，读者可以直接查阅相关的章节，协助完成目前的任务。由于本书的着眼点是理论和方法的含义解释和具体使用，因此，读者结合自己的工作阅读本书会有更深的体会。

本书由飞思科技产品研发中心策划，由刘寅琥主笔，参与本书编写的工作人员还有刘晶雯、陈海明、梁晔宁、吴小军、刘镇国、潘藏云、李小晶、任玲、黄学文和潘媛颖等，在此一并表示感谢！

由于作者水平有限，加之时间仓促，不足之处在所难免，敬请批评指正。

我们的联系方式如下：

咨询电话：(010) 68134545 68131648

电子邮件：support@fecit.com.cn

服务网址：http://www.fecit.com.cn http://www.fecit.net

通用网址：计算机图书、飞思、飞思教育、飞思科技、FECIT

飞思科技产品研发中心

绪言	1
本书的内容	1
本书如何使用	6
了解系统分析员	7
第1章 系统分析设计思想概述	29
1.1 软件分析方法的产生和发展	29
1.2 软件系统分析设计方法的基本概念	31
1.2.1 算法	32
1.2.2 数据结构	33
1.2.3 小结	38
1.3 系统分析的实现过程	38
1.3.1 信息处理的一般模式	39
1.3.2 信息处理的实际模式	41
1.4 系统分析和设计方法的内容和结构	41
1.4.1 系统分析和设计方法的内容	41
1.4.2 系统分析和设计方法的结构	44
1.4.3 系统分析和设计方法的学习方法	45
1.5 系统分析方法的比较	45
1.5.1 结构化程序设计 (SA)	47
1.5.2 面向对象 (OO)	48
1.5.3 泛型 (GP)	48
1.5.4 以组件为基础的设计 (CBD)	49
1.5.5 面向规则 (RO)	50
1.5.6 面向数据组织 (DO)	51
1.5.7 面向主题 (面向数据应用)	51
1.5.8 面向服务 (SO)	52
1.6 本章小结	53
第2章 基于变换: 结构化设计和面向数据的设计	55
2.1 软件的结构化分析设计	55
2.1.1 结构化程序	56
2.1.2 结构化定理	56
2.1.3 结构化系统分析设计过程——自顶向下逐步求精	59
2.1.4 面向数据流的设计方法	61
2.1.5 数据字典	61

2.1.6 模块与模块化	63
2.2 面向数据的分析与设计	65
2.2.1 典型的基于数据的应用系统	66
2.2.2 面向数据设计的核心问题	69
2.2.3 其他可能的问题	75
2.3 本章小结	81
第3章 基于分解：面向对象的机制与思想	83
3.1 面向对象思想概述	84
3.1.1 OO思想的本质是什么	85
3.1.2 面向对象系统对其思想的支持机制	86
3.1.3 OO的支持机制的实现	87
3.1.4 OO的好处和如何发挥这些优势	88
3.2 什么是对象	89
3.2.1 对象概念的引入	89
3.2.2 对象的静态结构	93
3.2.3 对象的空间结构	94
3.2.4 对象的时间结构(生命周期)	94
3.2.5 对象的环境	95
3.2.6 面向对象的系统分析和设计方法的目标	95
3.2.7 对象的实际生存环境和实现方式	96
3.2.8 小结	96
3.3 类	96
3.3.1 类的结构(静态结构)	96
3.3.2 类的机制	98
3.3.3 类之间的相互关系	110
3.3.4 接口	119
3.3.5 异常	122
3.3.6 RTTI和反射	125
3.3.7 包	126
3.3.8 组件和控件	127
3.3.9 CBD设计开发	130
3.3.10 消息(事件)	131
3.4 对象持久化	132
3.4.1 信息持久化的数据库方式	132
3.4.2 信息持久化的XML方式	160
3.5 本章小结	165

第 4 章 UML 的概念和使用	167
4.1 UML 概述	167
4.1.1 UML 是什么	168
4.1.2 产生背景	168
4.1.3 好处和使用方法	169
4.1.4 UML 工具介绍和使用方式	171
4.1.5 什么人使用 UML	173
4.1.6 UML 的总体架构	173
4.1.7 UML 的图和视图	174
4.1.8 UML 图的关系和由来	177
4.1.9 UML 的使用方式	178
4.2 用例图	179
4.2.1 用例和行为者的概念	180
4.2.2 行为者的确定	182
4.2.3 用例的提取	182
4.2.4 用例图	184
4.2.5 用例总结	185
4.2.6 实例	186
4.3 类图和对象图	192
4.3.1 基本标识	192
4.3.2 使用 UML 工具刻画类的结构	194
4.3.3 类之间的关系	195
4.3.4 包图	205
4.3.5 类图的作用	206
4.3.6 实例	207
4.4 顺序图	212
4.4.1 基本标识	212
4.4.2 作用	215
4.4.3 实例	215
4.5 协作图	224
4.5.1 基本标识	224
4.5.2 使用	227
4.6 状态图	227
4.6.1 主要标识符	228
4.6.2 实例	231
4.7 活动图	234

4.7.1	主要标识	235
4.7.2	实例	236
4.8	组件图	237
4.8.1	基本标识	237
4.8.2	组件图的作用	239
4.8.3	实例	239
4.9	物理图	240
4.9.1	基本标识	240
4.9.2	实例	242
4.10	UML 2.0 简介	243
4.11	本章小结	244
第 5 章	设计求精和设计模式	245
5.1	概述	246
5.1.1	使用面向对象思想的目标和设计模式	246
5.1.2	设计模式的概念	247
5.1.3	设计模式的原则	248
5.1.4	模式的运用	250
5.1.5	小结	251
5.2	创建型模式	251
5.2.1	抽象工厂 (AbstractFactory)	252
5.2.2	生成器 (Builder)	255
5.2.3	工厂方法 (Factory Method)	256
5.2.4	原型 (ProtoType)	258
5.2.5	单子 (Singleton)	260
5.3	结构型模式	262
5.3.1	适配器 (Adapter)	262
5.3.2	桥接 (Bridge)	263
5.3.3	装饰 (Decorator)	265
5.3.4	组合 (Composite)	267
5.3.5	外观 (Facade)	269
5.3.6	享元 (FlyWeight)	270
5.3.7	代理 (Proxy)	273
5.4	行为型模式	275
5.4.1	职责链 (Chain Of Responsibility)	275
5.4.2	命令 (Command)	276

5.4.3	解释器 (InterPreter)	279
5.4.4	迭代器 (Iterator)	280
5.4.5	中介者 (Mediator)	281
5.4.6	备忘录 (Memento)	283
5.4.7	状态 (State)	284
5.4.8	观察者 (Observer)	286
5.4.9	策略 (Strategy)	289
5.4.10	访问者 (Visitor)	291
5.4.11	模板 (Template Method)	294
5.5	OO 的其他开发模式与技巧	295
5.5.1	设计模式的提取	295
5.5.2	软件环境分析	296
5.5.3	Environment 类	297
5.5.4	接口、抽象类、类的完整实现层次	298
5.5.5	RAD 开发设计范式	298
5.6	模式的使用	303
5.6.1	模式使用的代价	303
5.6.2	逆模式化	306
5.7	本章小结	307
第 6 章	系统架构和系统模型	309
6.1	什么是系统架构、设计模型	310
6.1.1	设计模型是使用特定的观点、结构去看待分解问题	313
6.1.2	设计模型帮助提炼通用的功能模块	314
6.1.3	设计模型和信息模型与体系结构相关	314
6.1.4	设计模型的关键是便于应用和模型之间的映射	315
6.1.5	实现技术对使用设计模型的影响	315
6.2	ECC 模型	316
6.2.1	模型的核心	318
6.2.2	实体类的设计	319
6.2.3	集合类的设计	319
6.2.4	控制类的设计	320
6.2.5	ECC 模型小结	320
6.3	SLS 模型	321
6.3.1	模型的介绍	321
6.3.2	核心问题	323

6.4	层次模型	324
6.4.1	模型介绍	324
6.4.2	应用系统逻辑的层次模型	325
6.4.3	应用系统的层次计算架构	329
6.5	统一消息模型 (UM, 描述式交互模型)	339
6.6	应激反射-自适应模型	344
6.7	企业模型	348
6.8	通用开发模型	352
6.9	本章小结	357
第 7 章	设计演进和设计优化	359
7.1	系统分析的策略	361
7.1.1	软件目标	363
7.1.2	常用的思考方法和思考过程	365
7.1.3	系统的思考方法小结	368
7.2	设计分析和设计价值评估	368
7.2.1	设计平衡	369
7.2.2	系统的分布与系统的特征分析	380
7.3	设计演进	385
7.3.1	系统优化	387
7.3.2	系统的简化	389
7.3.3	设计演进的实例	391
7.4	常见设计方法	405
7.4.1	设计质量	406
7.4.2	系统约束	409
7.4.3	系统的生存环境	414
7.4.4	系统集成	416
7.4.5	系统设计方法和系统观察视角	424
7.5	本章小结	429
第 8 章	需求捕获和分析	431
8.1	软件需求概述	432
8.1.1	什么是软件需求	433
8.1.2	软件需求的结构	434
8.1.3	需求的生命周期	437
8.1.4	软件需求对于企业的意义	438
8.2	需求工程	438

8.2.1	需求工程概述	439
8.2.2	系统的可行性研究	444
8.2.3	客户访谈	446
8.2.4	确定需求	448
8.2.5	确定系统的功能点	450
8.2.6	确定项目约束	451
8.2.7	流程分析与组织机构分析	452
8.2.8	确定软件的使用方式	452
8.2.9	生成系统的软件需求分析说明书	453
8.3	需求驱动开发	453
8.3.1	需求对软件开发的影响	453
8.3.2	功能点形态分析	455
8.3.3	需求驱动下的软件开发	456
8.4	本章小结	458
第9章	系统分析员的工作职责与软件开发流程	459
9.1	软件开发过程概述	459
9.1.1	软件的生命周期	460
9.1.2	软件开发过程模型	461
9.2	软件开发过程中的不同角色	469
9.3	系统分析员的工作职责	471
9.3.1	概述	471
9.3.2	参与进行可行性研究	475
9.3.3	参与设计软件项目开发计划	475
9.3.4	需求捕获和分析	475
9.3.5	系统分析: 系统技术选型	476
9.3.6	系统分析: 需求满足的技术形态分析	479
9.3.7	系统分析: 确定和解决软件的难点	479
9.3.8	系统设计: 软件架构设计	480
9.3.9	系统设计: 系统功能流程设计	482
9.3.10	系统设计: 概要设计	482
9.3.11	系统设计: 详细设计	483
9.3.12	参与指定软件验收标准(一般包括测试用例)	483
9.3.13	简单的软件开发流程	484
9.3.14	软件项目总结	485
9.3.15	小结	486

9.4	软件开发的价值链和系统分析员的价值	487
9.5	理想的程序开发环境	490
9.5.1	支持软件开发过程	490
9.5.2	支持小组协作的软件开发	491
9.5.3	支持多个目标平台	492
9.5.4	支持建模 (UML 图制作) 与代码的同步	492
9.5.5	支持数据建模 (ERD 制作、XML 可视化定义编辑)	493
9.5.6	可视化的建模、需求捕获和确认	494
9.5.7	支持到目标环境的部署	494
9.5.8	其他的可能改变	495
9.5.9	小结	496
9.6	本章小结	496
第 10 章	后记	497
10.1	一般的问题分析思考方法	498
10.2	系统分析员的职业规划	500
10.3	系统分析员的知识更新和知识体系	506
10.4	尾声	511
附录 A	Java 语言简介	513
A.1	Java 语言的特点	513
A.2	Java 语法简介	513
A.3	Java 语言的优势简介	515
A.4	Java 技术体系	516

绪 言

欢迎进入系统分析员的世界！在今天，软件系统分析和设计能力，正在成为现代软件开发组织关注的重要技能。系统分析员在软件开发的过程中，正在发挥着越来越重要的作用。对于一个软件项目来说，好的系统分析和设计能够帮助软件开发组织高效率、高质量地进行软件开发。而对于软件质量的关注，已经成为现在的软件开发组织之间竞争的焦点。评价软件开发组织的能力已经不再局限于考察它能够完成什么软件，而更重要的是考察它如何更好地完成软件。这里的“好”包含了软件的质量、软件的开发成本、速度等各方面因素。通过好的系统分析和设计可以帮助软件开发组织达到这些目标。

质量成为软件开发的核心是与信息技术的发展息息相关的。随着信息技术的发展，软件的功能日益强大，应用的范围不断扩展，软件系统肩负的责任也因为软件应用领域的广泛而日趋重大。由软件错误引发的事故、灾难时有发生，并且后果更加严重。此外，随着硬件性能一日千里的提升，使得人们对于软件的要求，在大多数情况下不再单纯专注于软件基本任务的完成或处理性能的提升，而是综合的软件质量的提高。这些因素的共同作用使得对于软件质量的关注成为现代软件开发的一大特点。也正是由于这些变化，使得软件的系统分析，由原来只是一些对于软件的简单思考、设计，以及相关的实现技巧，转变成了一门不断发展的科学，并且逐渐产生了一类专门进行系统分析设计的软件开发人员——系统分析员，来负责有关的工作。目前，软件行业的先进开发组织借助各种先进的软件工具对系统进行深入的考察和分析，并在此基础上依赖分析员掌握的系统分析知识和经验完成设计，再借助管理良好的软件开发团队实现软件的开发。通过这种方式，可以高效率、高质量地进行软件开发，以满足今天各个行业对于软件质量的苛刻要求。

对于软件开发来说，达到高质量和高效率需要良好的系统分析设计和实现，这是软件开发过程中的两个核心因素。前者提供正确的前进目标，而后者则保证组织确实协调地向着设定的目标高质量地前进。这就意味着好的设计加上好的实现等于好的软件。本书主要讲解的就是系统分析方面的知识和技能，这些知识和技能用来帮助系统分析员完成良好的软件设计。

本书的内容

系统分析设计是软件开发组织开发优秀软件系统的重要工作，需要依靠拥有扎实理论基础和丰富实际工作经验的人来完成。本书内容是结合笔者的工作实践，讲解系统分析原理的理论、方法和实际操作方式。所有的理论和规范都进行了细致的解释，力求帮助读者正确地理解，在头脑中建立良好的理论和方法体系结构，而不是仅仅知道理论和规范。而且，本书的解释尽可能通俗易懂，以保证即使是缺乏相关经验和背景的读者也可以很好地理解，这是本书写作的重要目标。同时，理论与实践紧密结合，力求用理论来指导实际工作也是本书的



努力方向。

本书是一本面向系统分析员的书籍。主要讲解了系统分析员应该了解的理论和方法，并且对于系统分析员的工作职责进行了详细说明。本书主要介绍了以下几个部分。

1. 系统分析设计思想概述

系统分析和设计思想是系统分析员头脑中的指导思想。针对不同的软件问题，人们发展了不同的系统分析和设计的方法和原则。但是，这些方法实际都遵从着一些共同的规律，有一些基本的思考出发点，它们之间也有着很多继承和关联的关系，形成了完整的方法体系。了解这些系统分析方法的特点，掌握这些方法的原理对于系统分析员提高水平来说是非常重要的。实际上，任何一个软件都会混合使用不同的设计方法来进行设计，然后选择相应的机制实现。因此，这里会对主要的系统分析思想进行一个综述，帮助读者建立一个总体的印象，为后面深入地了解主要的系统分析设计思想打下基础。

2. 结构化分析设计方法和面向数据的分析设计方法

结构化分析设计方法是体现系统分析员技术素质的重要方面。虽然今天面向对象已经成为软件开发的主流，但是结构化分析方法仍然是系统分析的重要方法。在一些领域中，特别是与硬件结合紧密，或是对于性能有苛刻要求的领域，结构化分析设计方法仍然是主要的系统分析和设计方法。更何况面向对象的方法是在结构化方法的基础之上发展而来的，很多的分析设计思想，在两种方法之间是共享的。

除了结构化设计之外，这部分还会讲述面向数据的系统分析和设计方法。数据库系统是大多数信息系统都需要使用的数据库维护和管理系统，关系数据库又是技术成熟、使用方便的信息保存和检索方式。以数据库为核心进行系统设计，围绕数据模型，对数据表的访问等问题都可以建立一个以数据为中心的信息系统。在这一部分会描述这种系统设计思想和设计方法需要解决的主要问题。

3. OOAD 的主要原理

系统的分析和设计可以为软件系统绘制蓝图，建立目标，并且指出一条通往目标的切实可行的道路。对大型、复杂的软件系统而言，这个工作尤为重要。对于这样的系统，现在的软件分析和设计大多都采用面向对象的系统分析设计方法。这种方法特别适合支持面向对象语言机制的软件开发工具。新的编程语言也大部分都提供对面向对象机制的直接支持，并且把这作为工具或者语言的主要优势和特点加以发展。因此，在系统分析员的实际工作中，最为常用的技能就是使用面向对象的思想和方法进行系统分析和设计，也就是 OOAD (Object Oriented Analyze and Design)。和其他的系统分析设计方法相比较，面向对象的方法有其鲜明的技术和应用特点。只有深入地理解面向对象的机制，才能够正确使用这一思想，高质量地进行系统的分析和设计。

这一部分的总体目标就是相对深入地讲述 OO 的思想和原理，阐述各种面向对象的机制及该机制的正确使用方法。同时，提供如何使用这些机制的实际建议和方法。这一部分将是后面全部内容的基础，无论是 UML 还是设计模式，甚至是软件开发过程管理的方法都与面向对象的分析设计思想和基于此的开发方式密切相关。

4. UML 主要用法及常用 UML 工具

UML 是统一建模语言 (Unified Modeling Language) 的英文缩写, 是 20 世纪末才成为标准的一种基于面向对象思想的建模语言, 用来描述分析员建立的系统分析和设计模型。事实上现在已经成为软件系统分析建模领域的标准语言。UML 用来描述系统分析和设计模型, 使原本抽象的东西具体化、形式化, 便于分析员开展分析和设计工作, 并且可以帮助分析员之间进行交流。此外, 这一部分还将讲解主流的 UML 工具, 这是以往的 UML 教学里面经常忽略的部分。实际上, UML 工具在软件开发过程中起到了很重要的作用, 甚至正成为新的开发模式的核心支持工具, 更是系统分析员必须掌握的软件工具。因此, 学习 UML 就必须同时掌握相应的软件工具。

综上所述, UML 与系统分析员的工作息息相关, 就像编程语言对于程序员的作用一样。这一部分的内容, 对于系统分析员具有很强的实用性。

5. 设计模式

软件开发中的重用, 一直是人们进行软件开发、战胜软件危机的一种有效方式。重用包括了代码的重用、思想的重用、设计的中用, 以及资源的重用等不同方式。设计模式是以往成功设计思路的重用, 体现了软件重用的思想, 有助于提高软件设计的效率。同时, 通过一个个经过验证的成功的局部设计, 可以提高总体设计的质量。

设计模式是近年来系统分析领域中一个非常活跃的分支, 不仅体现了专家的经验, 而且可以把这些经验加以提炼总结, 使之模块化、形式化, 以便于今后的检索和使用。本书将讲解那些典型的设计模式, 除了介绍经典设计模式之外, 还提供了学习和使用模式的方法, 并特别讲解了设计模式的解释和含义。因为, 对于设计模式来说, 理解其含义和思想是灵活运用的基础。除此之外, 本书还会介绍一些行之有效的设计技巧, 这些同样可以帮助读者解决实际中的问题。

6. 软件架构和系统模型

软件的架构是软件设计的重要部分, 对于大型的软件系统来说尤其如此。同时, 进行软件架构的设计也是系统分析员的重要职责。软件架构的重要性及相关的讨论几乎发展成为一个独立的领域, 并且从系统分析员里分化出软件架构工程师, 来专门进行软件架构的设计。良好的软件架构是建立灵活的软件系统的重要保证。也就是说, 系统分析员应该认识到软件的灵活性来自于软件架构的灵活性, 而与编码的技巧关系不大。同样, 软件的安全性、高效性等几乎所有的质量因素都必须在软件的架构里仔细考虑、精心设计, 以便大大降低编码实现的风险, 使得软件的开发更加可控且更有保证。此外, 设计良好的软件架构, 还可以帮助软件开发人员快速领会系统, 有助于改善软件的质量。

系统模型是系统分析员对于信息系统建立的模型, 一般都是针对某类特定的应用。每一个软件架构实际上都是一个系统模型的实例, 系统模型是架构的模板。系统模型可以指导软件的设计, 避免出现设计架构时的多样性困难, 而这正是面向对象的系统分析和设计中的主要困难。



7. 设计优化与设计平衡

软件的设计不仅局限于实现指定的功能，或者说是完成用户的需求，更重要的是能够更好地满足用户的需求，并且降低软件开发、使用的风险，以及适应技术和未来应用的变化。系统分析员的价值，并不在于能够设计出软件，而是在于能够设计出结构良好、适应环境的软件。在这其中设计优化工作是一个核心。所有的这一切，都需要系统分析员综合考虑各种因素，在一系列平衡的原则指导之下，进行不断的调整和决策，设计出最终的软件系统。

软件系统的优化和设计平衡是进行系统设计工作的重要步骤，这一部分将讲述一些设计的原则、优化设计方法，帮助系统分析员从不同的侧面、不同的角度来评估和优化系统设计。

8. 需求捕获和分析

对于系统分析员来说，需求分析是一个颇具特色的工作。在这项工作中，系统分析员要面对的是对计算机未必非常了解的客户，以及未必非常了解的现实的应用领域。在这样的环境中，如何使得双方能够互相理解，并对最终产生的软件需要完成哪些功能达成共识，就是需求捕获和分析的核心任务。几乎每一个软件开发都了解需求分析的重要性，也都感受过需求分析的困难。这一部分就会对这个技术与业务相互衔接的特殊阶段进行深入剖析，帮助系统分析员解决相关的问题。

9. 系统分析员的工作职责

前面讲述的都是一些知识和技能，仅仅了解这些仍然算不上是一个好的系统分析员。这些知识和技能必须要结合到实际的软件开发工作中去。在这一部分，将介绍一些典型的软件开发过程，同时介绍在软件开发过程中，系统分析员的职责及其需要完成的工作。通过这些工作和步骤，可以帮助系统分析员融入到软件开发的实践中去。

10. 这些部分之间的关系小结

和任何一部依据逻辑组织内容的书籍一样，本书的内容也是按照系统分析员需要掌握的各知识点之间彼此的逻辑结构进行组织的。实际上，本书所介绍的内容分为4个大部分，每一部分都对应了一种软件开发过程中的具体角色。

首先，前面的第1章到第5章介绍的是程序设计方法，也就是单个的软件系统、软件模块如何实现。这部分的内容对应了软件开发组织里面的程序设计师或者叫系统设计师的知识领域，是他们需要掌握的知识内容。

接下去的第6章和第7章介绍了软件体系架构方面的内容，主要是面向系统架构师的。他们负责软件系统的结构设计，确定系统由哪些结构组成、系统的功能如何进行组织，以及相互之间的关系等。一般而言，架构师一样需要了解软件设计师的知识。但是架构师使用的思想和方法与程序设计方法是相互独立的。也就是说，这类角色的人员无论使用哪种程序设计方法都是可以的，架构设计与程序设计方法无关。这也区别了架构师和设计师的工作职责。

第8章则主要介绍了软件需求的分析和确定，这是属于需求分析师完成的内容。需求分析师既要掌握系统分析设计和实现的知识，又要具备良好的了解领域知识的方法和技能，以便顺利完成制订软件目标的任务。需求分析员的知识之所以放在系统分析员技能介绍的最