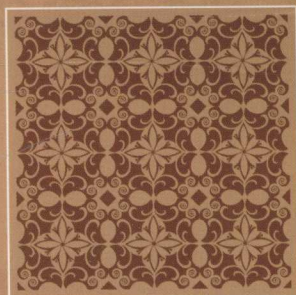


主编 窦万峰
参编 李亚楠 潘媛媛 林燕平

软件工程 方法与实践

第3版



Software Engineering
Theory and Practice (Third Edition)



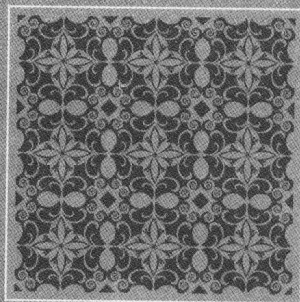
机械工业出版社
China Machine Press

高等院校计

主编 窦万峰
参编 李亚楠 潘媛媛 林燕平

软件工程 方法与实践

第3版



*S*oftware Engineering
Theory and Practice (Third Edition)



机械工业出版社
China Machine Press

图书在版编目 (CIP) 数据

软件工程方法与实践 / 窦万峰主编. —3 版. —北京: 机械工业出版社, 2016.10
(高等院校计算机课程案例教程系列)

ISBN 978-7-111-54948-2

I. 软… II. 窦… III. 软件工程—高等学校—教材 IV. TP311.5

中国版本图书馆 CIP 数据核字 (2016) 第 233984 号

本书分别从传统的结构化软件开发方法学和面向对象软件开发方法学两个方面介绍软件工程的理论和方法, 并将其融入实践, 通过丰富的案例介绍软件分析与设计方法及其模型, 深入讲解软件开发各个阶段的技术、方法和管理过程, 主要内容包括: 软件工程基础, 结构化分析、设计与测试, 面向对象分析、设计与测试, 软件维护与项目管理。

本书适合作为高等院校软件工程课程的教材, 也可作为软件开发从业人员的参考书。

出版发行: 机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码: 100037)

责任编辑: 曲 熠

责任校对: 董纪丽

印 刷: 北京瑞德印刷有限公司

版 次: 2016 年 10 月第 3 版第 1 次印刷

开 本: 185mm×260mm 1/16

印 张: 20.25

书 号: ISBN 978-7-111-54948-2

定 价: 45.00 元

凡购本书, 如有缺页、倒页、脱页, 由本社发行部调换

客服热线: (010) 88378991 88361066

投稿热线: (010) 88379604

购书热线: (010) 68326294 88379649 68995259

读者信箱: hzjsj@hzbook.com

版权所有·侵权必究

封底无防伪标均为盗版

本书法律顾问: 北京大成律师事务所 韩光 / 邹晓东

软件工程包含一系列软件开发的基本原理、方法和实践经验，用来指导人们进行正确的软件开发。软件工程强调从工程化的原理出发，按照标准化规程和软件开发实践来引导软件开发人员进行软件开发和实践活动，并进行过程改进，促进软件企业向标准化和成熟化的方向发展。软件工程是一门理论与实践相结合的学科，更注重通过实践来理解原理和方法。为此，我们结合多年的软件工程教学和项目开发经验，通过 5 个项目实例，从不同的角度、利用不同的方法来循序渐进地介绍软件开发过程中所涉及的原理、方法和技术。本书的另一个特色是从问题的角度引导学生根据自己的体会来讨论软件开发过程中的问题，进而理解软件工程的观念和原理，总结出一些有效的方法和实践经验。

编写思想

本书强调以问题为引导的软件工程所涉及的概念和方法，进而讨论具体的过程及其优缺点，并结合具体案例进行解析，让学生对问题产生的原因和新方法的提出有更深入的理解，还支持学生进行深入阅读。

我们将传统的结构化方法学和面向对象方法学分开介绍，这有利于学生理解二者的本质区别，厘清其分析与设计模型的不同特点，从而针对不同的项目来选择不同的开发方法学和过程。从结构化到面向对象的路线也便于学生逐步接受软件开发的思想和本质。

- 结构化方法学。重点讨论开发过程、原理和方法，这些都可以推广到面向对象的开发范型中。最后通过胰岛素输送这一高要求系统案例介绍如何将这些方法应用到实践中。
- 面向对象方法学。重点讨论面向对象分析模型和设计模型的构建，强调它们之间的关系，抓住面向对象模型开发的要点，通过 UML 建模语言来描述分析和设计模型，进一步加深学生对面向对象模型本质的理解，同时也清楚其适用的情况。最后通过 POS 机这一复杂系统案例帮助学生掌握面向对象分析与设计的主要思想。

本书还注重本科生研究性教学实践，针对现代软件开发方法——敏捷方法，重点介绍结对编程，在帮助学生理解结对编程思想的同时，分析其中存在的问题和解决方法，结合系统需求进行设计、实现与测试。通过这一过程可达到研究性教学的目的，也可将结对编程作为学期项目。

组织结构

本书分为四个部分，共 14 章内容。第一部分“软件工程基础”（第 1~4 章）主要从软件危机引出软件工程的基本概念和基本原理，介绍软件开发的工程化思想和开发过程等。第二部分“结构化分析、设计与测试”（第 5~9 章）针对传统结构化的软件开发方法学，主要介绍其基本概念、分析与设计过程、分析与设计模型、软件测试原理和技术、高要求系统的分析与设计方法等。第三部分“面向对象分析、设计与测试”（第 10~12 章）将介绍面向对象方法学的基本概念、用例分析模型及其设计过程、面向对象分析与设计模型、面向对象的实现以及测试技术。第四部分“软件维护与项目管理”（第 13、14 章）主要介绍软件维护策略与方法、软件项目管理概念与原理、软件成本估算以及项目计划与管理。

案例

由于本书分别介绍了传统的结构化方法学和面向对象方法学两大体系，因此专门选择了

适合不同方法学的具有代表性的案例进行研究,以便读者能够深入理解其各自的优势。这些案例中既有简单常见的应用系统,如面对面结对编程系统和 ATM 系统;也有比较实用的系统,如 POS 机系统;还有一些稍微复杂的系统,如分布式结对编程系统和胰岛素输送系统。这些系统由简单到复杂,循序渐进,引导学生逐步理解系统的开发过程和关键问题。

面对面结对编程系统是一个辅助学生进行结对编程和学习的系统,该系统支持角色交换、信息统计和相容性分析等功能,克服了编程过程中的一些不便,如交换位置、相互干扰等。同时,该系统采用一台主机支持结对,还具有节约实验室建设费用等优点。

POS 机系统是电子收款机系统的简称,通过计算机来处理销售和支付信息。该系统包括计算机终端、条码扫描仪、现金抽屉、票据打印机等硬件以及支持系统运转的软件,能够为不同服务的应用程序提供接口。收银员通过条码扫描仪读取的或键盘输入的商品条码号来记录商品信息,系统自动计算销售总价。收银员通过系统能够处理支付,包括现金支付、信用卡支付和支票支付。经理通过系统能够处理顾客退货。

ATM 系统即自动柜员机系统,能够自动处理银行储户的各种业务,如取款、存款、转账、查询、修改密码等。ATM 软件系统使客户能够直接访问银行计算机完成交易,无需银行工作人员的介入。

分布式结对编程系统支持跨地域的结对编程或学习。为了支持异地结对者像在本地一样方便地工作,系统通过文本、音频和视频进行交流。系统与集成开发环境进行集成,包括 VC++、Eclipse 等开发环境。系统支持角色交换,但通常不严格遵循“驱动者”和“领航者”的角色,所以分布结对编程的工具应该允许合作者很容易地访问控制键盘。

胰岛素输送系统是关于人体胰腺操作(一种体内组织)的仿真,其目标是帮助那些糖尿病患者控制血糖水平。该系统用于监控血糖浓度,根据需要输送正确剂量的胰岛素,对安全性的要求非常高。

意见与反馈

本书第 1~4 章由窦万峰编写,第 5~9 章由窦万峰和林燕平编写,第 10~13 章由窦万峰和李亚楠编写,第 14 章由窦万峰和潘媛媛编写。全书由窦万峰统稿、校对。

由于作者水平有限,因此难免有疏漏之处,恳请各位读者指正,意见可发至邮箱 douwfly@163.com。尤其是关于书中所选案例的详细程度和多样性,请读者多提意见,以便以后进行改进和完善。

作者

2016 年 9 月

推荐阅读



软件工程概论 第2版

作者：郑人杰 等 ISBN: 978-7-111-47821-8 定价：45.00元



面向对象分析与设计 第2版

作者：麻志毅 ISBN: 978-7-111-40751-5 定价：35.00元



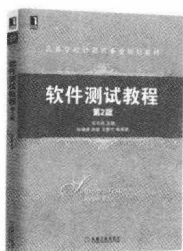
软件需求工程 第2版

作者：毋国庆 等 ISBN: 978-7-111-41735-4 定价：35.00元



软件项目管理案例教程 第3版

作者：韩万江 等 ISBN: 978-7-111-50163-3 定价：49.00元



软件测试教程 第2版

作者：宫云战 等 ISBN: 978-7-111-53270-5 定价：45.00元

推荐阅读



软件工程：实践者的研究方法（第7版）

作者：（美）Roger S. Pressman

译者：郑人杰 等

ISBN: 978-7-111-33581-8

定价：79.00元



软件工程：架构驱动的软件开发

作者：[美] 理查德 F. 施密特

译者：江贺 等

ISBN: 978-7-111-53314-6

定价：69.00元



软件可靠性方法

作者：（以色列）Doron A. Peled

译者：王林章 等

ISBN: 978-7-111-36553-2

定价：45.00元



IT项目管理（原书第7版）

作者：[美] 凯西·施瓦尔贝

译者：邢春晓 等

ISBN: 978-7-111-50956-1

定价：79.00元



软件建模与设计：UML、用例、模式和软件体系结构

作者：（美）Hassan Gomaa

译者：彭鑫 等

ISBN: 978-7-111-46759-5

定价：85.00元

目 录

前言

第一部分 软件工程基础

第1章 软件工程概述	2
1.1 引言	2
1.2 什么是软件	2
1.2.1 软件的定义与特性	2
1.2.2 软件技术的演化	3
1.3 什么是软件工程	4
1.3.1 软件危机	4
1.3.2 解决软件危机的途径	5
1.3.3 软件工程的定义	5
1.4 软件工程的基本原理与基本原则	7
1.4.1 基本原理	7
1.4.2 基本原则	8
1.5 软件工程开发方法学	8
1.5.1 结构化开发方法学	9
1.5.2 面向对象开发方法学	9
1.5.3 重型软件工程与轻型软件工程	10
1.6 小结	10
习题	11
第2章 软件过程	12
2.1 引言	12
2.2 什么是软件过程	12
2.2.1 软件过程的定义	12
2.2.2 软件过程框架	13
2.3 软件产品与过程	15
2.4 软件生存周期	17
2.5 软件工程活动	18
2.6 小结	20
习题	21
第3章 软件过程模型	22
3.1 引言	22

3.2 什么是软件过程模型	22
3.3 传统的软件过程模型	22
3.3.1 瀑布模型	22
3.3.2 增量模型	24
3.3.3 螺旋模型	25
3.4 面向对象模型	26
3.4.1 构件集成模型	26
3.4.2 统一过程模型	27
3.5 小结	29
习题	29

第4章 敏捷软件开发方法	30
4.1 引言	30
4.2 敏捷软件开发过程	30
4.2.1 敏捷过程	30
4.2.2 敏捷开发原则	31
4.3 Scrum 开发过程	31
4.3.1 Scrum 的特点	31
4.3.2 Scrum 模型与过程	32
4.4 极限编程	33
4.4.1 什么是极限编程	33
4.4.2 极限编程的要素	34
4.5 结对编程	35
4.5.1 什么是结对编程	36
4.5.2 结对编程的优势分析	37
4.5.3 结对编程的分类	39
4.5.4 结对编程的方式	39
4.6 小结	41
习题	41

第二部分 结构化分析、设计与测试

第5章 软件需求分析	44
5.1 引言	44
5.2 什么是软件需求	44
5.3 需求分析过程	45

5.4 会谈技术	49	7.10.2 编码风格	102
5.4.1 非正式会谈	49	7.11 小结	103
5.4.2 正式会谈	49	习题	103
5.5 调查技术	50		
5.5.1 确定调查内容	50	第 8 章 结构化软件测试	106
5.5.2 可靠可信分析	51	8.1 引言	106
5.6 场景分析技术	51	8.2 软件测试的目的和原则	106
5.7 小结	52	8.3 软件测试的基本过程	108
习题	52	8.3.1 单元测试	108
		8.3.2 集成测试	110
第 6 章 结构化分析	53	8.3.3 确认测试	110
6.1 引言	53	8.3.4 系统测试	111
6.2 结构化分析模型	53	8.4 测试用例设计	111
6.3 面向数据流的建模方法	54	8.5 黑盒测试技术	112
6.3.1 数据流建模方法	54	8.5.1 等价类划分	112
6.3.2 实例分析	57	8.5.2 边界值分析	113
6.4 面向数据的建模方法	58	8.5.3 错误推测	114
6.4.1 数据建模方法	58	8.5.4 因果图	114
6.4.2 实例分析	59	8.6 白盒测试技术	117
6.5 面向状态的建模方法	60	8.6.1 逻辑覆盖	117
6.5.1 状态建模方法	60	8.6.2 路径覆盖	120
6.5.2 实例分析	61	8.6.3 循环路径测试策略	122
6.6 规格说明书编写示例	62	8.7 集成测试技术	123
6.7 小结	70	8.7.1 集成策略	123
习题	70	8.7.2 性能测试	125
		8.7.3 实例分析	127
第 7 章 结构化设计	71	8.8 调试技术	133
7.1 引言	71	8.8.1 调试过程	133
7.2 软件设计过程	71	8.8.2 调试策略	134
7.3 软件模块化设计	75	8.9 软件测试文档	135
7.4 软件结构	78	8.9.1 软件测试计划文档	135
7.5 结构化概要设计	79	8.9.2 集成测试文档	137
7.5.1 数据流模型	79	8.10 小结	139
7.5.2 数据流设计方法	80	习题	139
7.5.3 实例分析	81		
7.6 概要设计文档编写示例	83	第 9 章 高要求系统的分析与设计	140
7.7 详细设计	90	9.1 引言	140
7.8 结构化详细设计	91	9.2 什么是高要求系统	140
7.9 详细设计文档编写示例	98	9.3 高要求系统的需求分析	142
7.10 编码实现	101	9.3.1 风险需求描述	143
7.10.1 编码语言	101	9.3.2 安全性描述	146

9.3.3 信息安全描述	146
9.3.4 软件可靠性描述	146
9.3.5 胰岛素输送系统完整的需求 描述	147
9.4 形式化描述方法	149
9.4.1 软件过程中的形式化描述	149
9.4.2 接口描述方法	150
9.4.3 行为描述	153
9.5 高要求系统的设计	155
9.6 高要求系统的开发	156
9.7 系统验证	160
9.7.1 可靠性验证	160
9.7.2 安全性保证	161
9.7.3 信息安全评估	164
9.8 小结	165
习题	166

第三部分 面向对象分析、 设计与测试

第 10 章 面向对象分析	168
10.1 引言	168
10.2 面向对象模型	168
10.3 UML	170
10.3.1 UML 的组成	170
10.3.2 UML 的视图	171
10.4 面向对象分析过程	172
10.5 用例驱动分析	173
10.5.1 用例建模分析	173
10.5.2 开发活动图	177
10.5.3 开发泳道图	178
10.6 领域与业务建模	179
10.6.1 识别业务类和领域类	179
10.6.2 业务类图	180
10.6.3 识别属性和操作	181
10.6.4 开发协作图	182
10.6.5 开发包图	182
10.6.6 逻辑架构	183
10.7 系统行为建模	185
10.7.1 系统顺序图	185
10.7.2 建立操作契约	186

10.7.3 建立顺序图	187
10.7.4 系统状态图	188
10.8 POS 机系统案例分析	188
10.9 分布式结对编程系统分析	191
10.9.1 项目概述	191
10.9.2 功能描述	192
10.9.3 逻辑分析与建模	200
10.10 小结	204
习题	205

第 11 章 面向对象设计	206
11.1 引言	206
11.2 面向对象设计模型	206
11.3 构件设计	207
11.3.1 构件设计的步骤	207
11.3.2 构件设计的原则	208
11.4 并发性设计	209
11.5 设计模式	211
11.5.1 基于职责的设计	211
11.5.2 常见的设计模式	212
11.6 面向对象详细设计	215
11.6.1 模型精化	216
11.6.2 逻辑架构精化设计	219
11.6.3 分层设计	221
11.6.4 类操作设计	223
11.7 方法设计	229
11.8 精化设计	234
11.9 数据存储与持久性设计	238
11.10 部署设计与构件图	243
11.11 小结	244
习题	245

第 12 章 面向对象实现与测试	246
12.1 引言	246
12.2 面向对象实现	246
12.3 POS 机系统实现	247
12.4 分布式结对编程系统实现	253
12.5 面向对象测试	256
12.6 面向对象测试策略	258
12.7 测试驱动开发	261
12.7.1 什么是测试驱动开发	261

12.7.2 测试驱动开发的步骤 264

12.7.3 编写测试程序 266

12.7.4 代码重构 270

12.8 小结 271

习题 271

第四部分 软件维护与项目管理

第 13 章 软件维护 274

13.1 引言 274

13.2 软件的可维护性 275

13.3 软件维护的类型 275

13.4 软件维护方法 277

13.5 提高软件的可维护性 279

13.5.1 结构化维护与非结构化维护 279

13.5.2 提高软件可维护性的技术途径 280

13.6 小结 281

习题 281

第 14 章 软件项目管理 283

14.1 引言 283

14.2 软件项目组织 285

14.3 软件过程管理 287

14.3.1 软件过程度量 287

14.3.2 软件过程改进 288

14.3.3 软件项目度量 289

14.4 软件项目风险管理 289

14.4.1 风险识别 289

14.4.2 风险预测 290

14.4.3 风险管理 290

14.5 软件配置管理 291

14.5.1 基本概念 291

14.5.2 软件配置管理过程 292

14.6 软件项目估算 293

14.6.1 软件项目资源 293

14.6.2 软件规模度量 294

14.6.3 估算管理 296

14.7 分解技术 296

14.7.1 基于问题分解的估算 296

14.7.2 基于过程分解的估算 298

14.8 经验估算技术 299

14.8.1 专家类比推断 299

14.8.2 中级 COCOMO 估算模型 300

14.9 软件质量管理 303

14.9.1 软件质量保证 303

14.9.2 软件质量度量 305

14.10 项目进度管理 308

14.10.1 项目进度管理计划 308

14.10.2 进度安排 309

14.10.3 进度跟踪管理 311

14.11 小结 311

习题 312

参考文献 314

软件工程基础

本部分将介绍软件工程的基本概念、软件过程及其模型和敏捷软件开发方法，包括软件工程概述、软件过程、软件过程模型和敏捷软件开发方法四章内容，将关注以下问题：

- 软件工程的定义。
- 软件开发工程化思想。
- 软件工程的基本原理和基本原则。
- 软件过程。
- 软件过程模型。
- 常见的软件过程模型。
- 敏捷软件开发方法。

学过本部分内容后，请思考下列问题：

- 如何选择软件过程模型？
- 为什么统一过程模型得到广泛流传？
- 敏捷过程有哪些优势？其模型对现代软件开发产生了什么影响？
- 软件工程有哪些实践活动？
- 如何实施结对编程？

软件工程概述

1.1 引言

软件工程（Software Engineering, SE）是在 20 世纪 60 年代末期提出的。提出这一概念的目的在于倡导以工程化的思想、原则和方法开发软件，并用来解决软件开发和维护过程中出现的诸多问题。

1.2 什么是软件

既然软件工程的主角是软件开发，那么在现代社会中，软件担任的究竟是一种什么样的角色呢？我们使用的大部分软件同时担任着两个角色，既是软件产品，又是软件工具。软件产品是指为最终用户使用并带来益处的具有商业价值的软件系统。软件工具是指开发其他软件的软件系统。我们可以利用这些软件系统存储信息或进行信息的变换等。

1.2.1 软件的定义与特性

什么是软件？软件是计算机系统中与硬件相对应的另一部分，是一系列程序、数据及其相关文档的集合。程序、数据和文档称为软件的三要素，如图 1-1 所示。

- 程序是按照特定顺序组织的计算机数据和指令的集合。
- 数据是使程序能正常执行的数据结构。
- 文档是与程序的开发、维护和使用有关的资料。

计算机软件的核心是程序，而文档则是软件不可分割的组成部分。

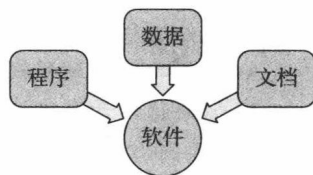


图 1-1 软件的三要素

要理解软件的真实含义，需要了解软件有哪些特征。与软件相对应的是硬件，在计算机的体系结构中，人们当初利用智慧创造的硬件是有物理形态的。现在，人们利用结构化的思想创造出的软件是逻辑的而不是有固有形态的实体，所以，计算机软件和硬件有着截然不同的特征。

复杂性。软件是一个庞大的逻辑系统，比人类构造的其他产品更复杂，甚至硬件的复杂性和软件比起来也是微不足道的。此外，软件主要是依靠人脑的“智力”构造出来的，多种人为因素使得软件难以统一化，更增加了其复杂性。软件的复杂性使得软件产品难以理解、难以生产、难以维护，更难以对生产过程进行管理。

一致性。软件必须和运行它的硬件保持一致，这是由软件对硬件的依赖所决定的，一般采用软件顺应硬件接口，而不是硬件顺应软件的方案。如果硬件系统是“现存”的，软件必须和现有硬件系统接口保持一致。此外，由于计算机的软件和硬件具有功能互换性，所以也可能出现用软件来替代硬件接口的功能。

退化性。软件与硬件相比不存在磨损和老化的问题。事实上,软件不会磨损,但它会因缺陷和过时而退化,因此,软件在其生命周期中需要进行多次的维护,直至被淘汰。

易变性。软件在生产过程中,甚至在投入运行之后,也可以再改变。软件必须能够经历变化并容易改变,这也是软件产品的特有属性。软件易变性的好处是:改变软件往往可以收到改变或者完善系统功能的效果;修改软件比更换硬件容易,使得软件具有了易维护、易移植、易复用的特征。但这种动态的变化难以预测且难以控制,可能对软件的质量产生负面影响。

移植性。软件的运行受计算机系统的影响,不同的计算机系统平台可能会导致软件无法正常运行,即软件的移植性。好的软件在设计时就要考虑到软件如何应用到不同的系统平台。

高成本。软件的开发是一个复杂的过程,需要耗费大量的开发成本和管理成本,导致软件的成本比较高。

1.2.2 软件技术的演化

软件的发展经历了一个演化的过程,自从20世纪40年代产生了世界第一台计算机后,伴随而生的就是程序或软件。软件的演化大致经历了以下四个阶段。

第一阶段。1946年到20世纪60年代初是计算机软件发展的初期,一般称为程序设计阶段,其主要特征是程序生产方式为个体手工方式。

第二阶段。20世纪60年代初到70年代初是计算机软件发展的第二个阶段,也称为程序阶段。在这个阶段,软件工程学科诞生了。程序的规模已经发展得很大了,软件开发需要多人分工协作,软件的开发方式由个体生产发展为小组生产。但是,由于小组生产的开发方式基本上沿用了软件发展早期所形成的个体化的开发方式,软件的开发与维护费用以惊人的速度增加,导致许多软件产品后来根本不能维护,最终导致软件危机的出现。

第三阶段。20世纪70年代中期至80年代中期是计算机软件发展的第三个阶段,一般称为软件工程阶段。在这个阶段,软件工程师把工程化的思想加入软件的开发过程中,用工程化的原则、方法和标准来开发和维护软件。

第四阶段。从20世纪80年代中期至今,面向对象的方法学受到了人们的重视,促进了软件业的飞速发展,软件产业在世界经济中已经占有举足轻重的地位,这个阶段一般称为面向对象阶段。

20世纪末开始流行的Internet给人们提供了一种全球范围的信息基础设施,形成了一个资源丰富的计算平台,未来如何在Internet平台上进一步整合资源,形成巨型的、高效的、可信的虚拟环境,使所有资源能够高效、可信地为所有用户服务,成为软件技术的研究热点。

Internet平台具有一些传统软件平台不具备的特征:分布性、结点的高度自治性、开放性、异构性、不可预测性、连接环境的多样性等。这对软件工程的发展提出了新的问题,软件工程需要新的理论、方法和技术来应对这个问题。目前投入很大精力研究的中间件技术就是这方面的典型代表。Internet和基于Internet应用的快速发展与普及,使计算机软件所面临的环境开始从静态封闭逐步走向开放、动态和多变。软件系统为了适应这样一种发展趋势,将会逐步呈现出柔性、多目标、连续反应式的网构(NetWare)软件系统的形态。

随着Internet的发展与应用,出现了“互联网+”的新概念。“互联网+”是创新2.0下的互联网发展的新业态,是知识社会创新2.0推动下的互联网形态演进及其催生的经济社会发展新形态。“互联网+”催生了一系列软件及其平台的需求。近几年来,“互联网+”已经改变了多个行业,当前大众耳熟能详的电子商务、互联网金融(ITFIN)、在线旅游、在线影

视、在线房产等行业都是“互联网+”的杰作。

随着宽带无线接入技术和移动终端技术的飞速发展，人们迫切希望能够随时随地乃至在移动过程中都能方便地从互联网获取信息和服务，移动互联网应运而生并迅猛发展。然而，移动互联网在移动终端、接入网络、应用服务、安全与隐私保护等方面还面临着一系列的挑战。其基础理论与关键技术的研究，对于国家信息产业整体发展具有重要的现实意义。

1.3 什么是软件工程

在软件开发的早期阶段，人们过高地估计了计算机软件的功能，认为软件能承担计算机的全部责任，甚至有些人认为软件可以做任何事情。如今，绝大多数专业人士已经认识到软件神化思想的错误。尤其是软件危机的出现，迫使人们思考一个问题，那就是软件并非万能的，难以满足人们各种各样的需求，需要提出有效的开发与维护方法来指导人们高效率地开发高质量的软件。

1.3.1 软件危机

计算机硬件技术的不断进步，要求软件能与之相适应。然而，软件技术的进步一直未能满足形势发展提出的要求，致使问题积累起来，形成了日益尖锐的矛盾，最终导致了软件危机。软件危机主要表现为：

- 软件的规模越来越大，复杂度不断地增加，软件的需求量也日益增大，且价格昂贵，供需差日益增大。
- 软件的开发过程是一种高密集度的脑力劳动，软件开发工作常常受挫，质量差，很难按照要求的进度表来完成指定的任务，软件的研制过程管理起来困难，往往失去控制。
- 软件开发的模式及技术已经不能适应软件发展的需要。因此，导致大量低质量的软件涌向市场，有些软件开发出来已远远超出了预算，有的软件甚至在开发过程中就夭折了。例如，伦敦股票交易系统当初预算 4.5 亿英镑，后来追加到 7.5 亿，历时 5 年，但最终还是失败，导致伦敦股票市场声誉下跌。

下面通过伦敦救护服务系统的例子来分析软件危机的表现和问题。

伦敦救护服务系统覆盖伦敦市区 600 平方千米的地域和大约 680 万的救护人口，是世界上最大的救护服务中心。该服务中心拥有 318 辆事故与应急救护车和 445 病人运输救护车、一个摩托车接应团队和一架直升机。中心的工作人员达到 2746 人，他们分布在伦敦市区 70 个救护站，每个救护站又分成 4 个运营部门。

伦敦救护服务系统的目的是提供自动化救护呼叫请求和处理紧急救护需要，通过计算机系统处理人工系统的所有任务。呼叫 999 和请求救护服务将呼叫者和派遣者连接起来，派遣者记录呼叫细节和分派合适的车辆。分派者将选择救护车并转发救护信息给车载系统。

伦敦救护服务系统包括 3 个组成部分：①计算机辅助派遣系统，包括软硬件基础设施、事故记录保存系统、无线电通信系统和无线电系统接口；②计算机地图显示系统，包括复杂地域地形分析软件；③自动化车辆定位系统，具有车辆自动定位能力，以便以最短的时间到达指定位置，并跟踪分析系统的性能。另外，伦敦救护系统还包括无线电系统和移动数据终端。

伦敦救护服务系统项目于 1987 年 4 月启动，前期投资 250 万英镑用于开发一个有限功能的派遣系统；1989 年设计规格被重新修改，增加了移动数据终端和声讯转换系统。1990

年10月项目经过两次峰值负载性能测试失败而被迫终止。截至项目被取消时为止,项目已经花费了750万英镑的费用,超过预算的300%。

1991年8月项目重新启动。为了保证项目的顺利进行,合作方定期举行会议来协调项目进度和解决存在的问题。但是截至1992年1月,项目还是被延期。派遣系统没有完全实现和测试,无线电接口系统未能按时交付,救护车数据终端设计和定位系统需求还需进一步完善,车载定位跟踪系统没有完成安装、调试。

1992年10月26日,整个新系统全部运转。但是过载问题仍然没有很好地解决,存在呼叫丢失和响应不及时问题。1992年10月27日,系统不得不改为半自动化方式。1992年11月,系统运行性能开始全面下降,并最终导致系统锁死。由于没有及时响应和系统存在的故障,导致病人死亡事件发生。工作人员试图切换和重启系统,但均告失败。由于系统没有备份系统,操作人员被迫恢复到完全人工过程。

伦敦救护服务系统的失败归因于一系列软件工程中的错误,特别是项目管理中的缺陷,从而导致了1992年秋天出现的两次故障。伦敦救护服务系统失败的例子告诉我们,系统的复杂性和庞大规模、系统需求的不准确和经常变更,以及管理不到位等因素是导致系统失败的主要原因。

我们称软件开发和维护过程中所遇到的严重问题为软件危机。软件危机主要是两个方面的问题:一是如何开发软件,以满足对软件日益增长的客户需求;二是如何维护数量不断膨胀的现有软件。

1.3.2 解决软件危机的途径

在软件危机相当严重的背景下,软件工程产生了。在引入工程化的思想后,人们总结了导致软件危机的原因,并提出了相应的解决对策。

在软件开发的初期阶段,需求提得不够明确,或未能得到确切的表达。开发工作开始后,软件开发人员和用户又未能及时交换意见,造成开发后期矛盾集中暴露。如果在开发的初期阶段需求不够明确,或未能得到确切的表达,工作人员不与客户及时地交换意见,就有可能导致软件开发后期的问题无法解决。如果仅仅认为软件的开发是编写程序,软件开发前期的需求分析不到位,很有可能使得后期开发的软件达不到客户的要求,导致软件的二次开发。

需求分析后,要做好软件定义时期的工作,这样可以在一定的程度上降低软件开发的成本,同时在无形中提高软件的质量,毕竟软件是一种商品,提高质量是软件开发过程中的重中之重。

开发过程要有统一的、公认的方法论和规范指导,参加的人员必须按照规定的方法进行开发。由于软件是逻辑部件,开发阶段的质量难以衡量与评价,开发过程的管理和控制较难,因此要求开发人员要有统一的软件工程理论来指导。

必须做好充分的检测工作,提交给客户高质量的软件。要借鉴软件开发的经验和积累的有关软件开发的数据,确保开发工作的计划按时完成,在期限内完成软件的开发。

1.3.3 软件工程的定义

关于软件工程的定义有许多,下面是流行的几种定义:

- B. W. Boehm 将软件工程定义为运用现代科学技术知识来设计并构造计算机程序及为开发、运行和维护这些程序所必需的相关文件资料。
- Fritz Bauer 将软件工程的定义为经济地获得能够在实际机器上有效运行的可靠软件而建立和使用的一系列完善的工程化原则。

- IEEE 软件工程标准术语对软件工程的定义为：软件工程是开发、运行、维护和修复软件的系统方法，其中“软件”的定义为计算机程序、方法、规则、相关的文档资料以及在计事机上运行时所必需的数据。

尽管软件工程的具体定义不尽相同，但其主要思想都在强调在软件开发的过程中需要应用工程化思想的重要性。工程化思想的核心是把软件看作一个需要通过需求分析、设计、实现、测试、管理和维护的工程产品。用完善的工程化原理研究软件生产的方法规范软件的开发，不仅保证软件开发在指定的期限内完成，而且可以节约成本，保证软件的质量。

软件工程是一门研究如何用系统化、规范化、数量化等工程化思想和方法进行软件开发、维护和管理学科。因此，软件工程学涉及的范围很广，涉及计算机科学、管理学、系统工程学和经济学等多个学科领域。

软件工程学分成软件开发方法和软件工程管理两个方面，重点是对软件开发方法和工程性技术的研究。软件开发技术和软件工程管理的复杂程度均与软件的规模密切相关。规模越大的软件产品，越要严格遵守软件工程的开发原则和方法。

软件开发不同于一般的产品生产，因为软件是一种没有具体形体和尺寸的特殊的产物，它创造的唯一产品或者服务是逻辑载体。它提供的产品或服务是逻辑的，具有独特性、临时性和周期性等特点。不同于其他产品的制造，软件过程更多的是设计过程。另外，软件开发不需要使用大量的物质资源，而主要是人力资源。并且，软件开发的产品只是程序代码和技术文件，并没有其他的物质结果。基于上述特点，与其他项目管理相比，软件项目管理有很大的独特性。

软件开发过程中除编写代码以外，还需要编写大量的文档和建立各种模型，需要耗费较多的时间与费用，且工作效率低下。因此，软件开发还需要大量的工具来提高开发效率，如文档编辑工具、代码编辑与调试工具、测试工具、建模工具等等。

综上所述，我们把方法、工具和过程称为软件工程的三要素，如图 1-2 所示。软件工程方法为软件开发提供了“如何做”的技术；软件工具为软件工程方法提供了自动的或半自动的软件支撑环境；过程是为了获得高质量的软件所需要完成的一系列任务框架，规定了完成各项任务的工作步骤。

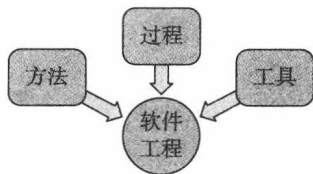


图 1-2 软件工程的三要素

软件开发工程化的思想主要体现在软件项目管理。软件项目管理的作用是，一方面可提高软件质量，降低成本；另一方面可为软件的工程化开发提供保障。与其他行业项目相比，软件行业的项目具有其特殊性。随软件行业的迅猛发展，一些问题和危机逐渐暴露出来。例如，项目时间总是推迟、项目结果不能令客户满意、项目预算成倍超出、项目人员不断流动等都是软件开发商不断面临的一些问题。

软件工程学家分析认为，导致上述情况的主要原因是缺乏软件过程控制能力，开发过程随心所欲，时间计划和费用估算缺乏现实的基础，产品质量缺乏客观基础，软件开发的成败建立在个人能力基础上等。

从商业的角度，软件也称为软件产品，客户必然会更新软件开发的质量、成本和工期。因此，软件工程管理三要素包括质量、成本和工期，如图 1-3 所示。质量包括质量定义、质量管理、质量保证、质量评价等。成本包括成本预算和核算、成本管理、资源管理等。工期包括工程进度管理、组织人员管理、工作量管理、配置管理等。

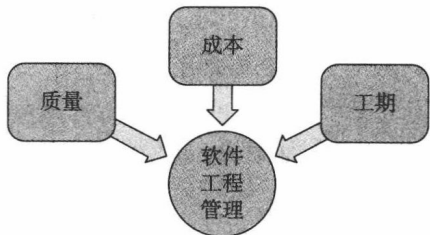


图 1-3 软件工程管理的三要素