



普通高等教育“十一五”国家级规划教材



21世纪大学本科
计算机专业系列教材

钱乐秋 赵文耘 牛军钰 编著

软件工程 (第3版)

<http://www.tup.com.cn>

- 根据教育部“高等学校计算机科学与技术专业规范”组织编写
- 与美国 ACM 和 IEEE CS *Computing Curricula* 最新进展同步
- 上海市精品课程主讲教材



清华大学出版社



普通高等教育“十一五”国家级规划教材

上海市精品课程主讲教材

21世纪大学本科计算机专业系列教材

软件工程

(第3版)

钱乐秋 赵文耘 牛军钰 编著

清华大学出版社

北京

内 容 简 介

本书系统地介绍了软件工程的观念、原理、过程及主要方法,内容上覆盖了 ACM 和 IEEE 制定的 Computing Curricula 中有关软件工程的主要知识点。本书在介绍软件工程的基本概念和基本原理的基础上,重点介绍软件开发方法和技术,包括经典的和常用的方法,如结构化方法、面向数据结构方法和面向对象方法,以及一些软件的新技术和新方法,如 UML 2.0、基于构件的开发、敏捷软件开发、Web 工程、CMM 和 CMMI 等。此外,本书尽量采用国标、ISO 标准及《计算机科学技术百科全书》对专业术语的名称及其语义解释,必要时,术语名称会同时给出其他习惯称谓。

本书适合作为高等学校计算机科学与技术学科、软件工程学科各专业的教材,也可作为软件开发人员的参考书。

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。

版权所有,侵权必究。侵权举报电话:010-62782989 13701121933

图书在版编目(CIP)数据

软件工程/钱乐秋,赵文耘,牛军钰编著. —3 版. —北京:清华大学出版社,2016

21 世纪大学本科计算机专业系列教材

ISBN 978-7-302-44608-8

I. ①软… II. ①钱… ②赵… ③牛… III. ①软件工程—高等学校—教材 IV. ①TP311.5

中国版本图书馆 CIP 数据核字(2016)第 175388 号

责任编辑:张瑞庆

封面设计:常雪影

责任校对:焦丽丽

责任印制:宋 林

出版发行:清华大学出版社

网 址: <http://www.tup.com.cn>, <http://www.wqbook.com>

地 址:北京清华大学学研大厦 A 座 邮 编:100084

社 总 机:010-62770175 邮 购:010-62786544

投稿与读者服务:010-62776969, c-service@tup.tsinghua.edu.cn

质 量 反 馈:010-62772015, zhiliang@tup.tsinghua.edu.cn

课 件 下 载: <http://www.tup.com.cn>, 010-62795954

印 装 者:北京嘉实印刷有限公司

经 销:全国新华书店

开 本:185mm×260mm 印 张:24.5 字 数:595 千字

版 次:2007 年 3 月第 1 版 2016 年 9 月第 3 版 印 次:2016 年 9 月第 1 次印刷

印 数:1~2000

定 价:44.90 元

产品编号:071521-01

第 1 章 概论	1
1.1 计算机软件	1
1.1.1 软件的发展	1
1.1.2 软件的特点	2
1.1.3 软件的分类	3
1.1.4 软件语言	3
1.2 软件工程	5
1.2.1 软件工程定义	5
1.2.2 软件工程框架	6
1.2.3 软件生存周期	7
1.3 软件过程	7
1.3.1 软件生存周期过程	8
1.3.2 能力成熟度模型	11
1.3.3 能力成熟度模型集成	13
1.4 软件过程模型	16
1.4.1 瀑布模型	16
1.4.2 演化模型	17
1.4.3 增量模型	18
1.4.4 原型模型	18
1.4.5 螺旋模型	20
1.4.6 喷泉模型	21
1.4.7 基于构件的开发模型	21
1.4.8 形式化方法模型	22
1.5 CASE 工具与环境	23
1.5.1 软件工具	23
1.5.2 软件开发环境	25
1.6 小结	26
习题	26

第2章 系统工程	27
2.1 基于计算机的系统	27
2.2 系统工程的任务	28
2.3 可行性分析	29
2.3.1 经济可行性	29
2.3.2 技术可行性	31
2.3.3 法律可行性	31
2.3.4 方案的选择和折衷	31
2.4 小结	32
习题	32
第3章 需求工程	33
3.1 需求工程概述	33
3.2 需求获取	35
3.2.1 软件需求	35
3.2.2 需求获取方法与策略	36
3.3 需求分析、协商与建模	40
3.3.1 需求分析原则	40
3.3.2 信息域	40
3.3.3 需求协商	41
3.3.4 需求建模	41
3.4 需求规约与验证	42
3.4.1 需求规约的原则	42
3.4.2 需求规约	42
3.4.3 需求验证	44
3.5 需求管理	44
3.6 小结	45
习题	45
第4章 设计工程	46
4.1 软件设计工程概述	46
4.2 软件设计原则	48
4.2.1 抽象与逐步求精	49
4.2.2 模块化	49
4.2.3 信息隐藏	50
4.2.4 功能独立	51
4.3 软件体系结构设计	53
4.3.1 体系结构发展过程	53

4.3.2	软件体系结构的风格	54
4.3.3	评估可选的体系结构	56
4.4	部件级设计技术	57
4.4.1	结构化程序设计方法	58
4.4.2	图形表示法	58
4.4.3	判定表	61
4.4.4	设计性语言 PDL	62
4.5	设计规约与设计评审	63
4.5.1	设计规约	63
4.5.2	设计评审	64
4.6	小结	65
	习题	65
第 5 章	结构化分析与设计	67
5.1	结构化分析方法概述	67
5.2	数据流图	69
5.2.1	数据流图的图形表示	69
5.2.2	分层数据流图的画法	72
5.3	分层数据流图的审查	76
5.3.1	分层数据流图的一致性和完整性	77
5.3.2	构造分层 DFD 时需要注意的问题	80
5.3.3	分解的程度	82
5.4	数据字典	82
5.4.1	字典条目的种类及描述符号	82
5.4.2	字典条目	83
5.4.3	字典条目实例	87
5.4.4	数据字典的实现	88
5.5	描述基本加工的小说明	88
5.5.1	结构化语言	89
5.5.2	判定表	90
5.5.3	判定树	91
5.6	结构化设计概述	91
5.6.1	结构图	92
5.6.2	启发式设计策略	94
5.6.3	结构化设计的步骤	96
5.7	数据流图到软件体系结构的映射	97
5.7.1	信息流	97
5.7.2	数据流图映射到结构图的步骤	98
5.7.3	变换分析	98

5.7.4	事务分析	101
5.7.5	分层 DFD 的映射	102
5.8	初始结构图的改进	104
5.8.1	结构图改进实例	104
5.8.2	结构图改进技巧	106
5.9	小结	108
	习题	108
第 6 章	面向数据结构的分析与设计	110
6.1	JSP 方法	110
6.1.1	数据结构与程序结构的表示	110
6.1.2	JSP 方法的分析和设计步骤	114
6.2	JSD 方法简介	118
6.3	小结	119
	习题	119
第 7 章	面向对象方法基础	120
7.1	面向对象的基本概念	121
7.2	面向对象分析和设计过程	124
7.2.1	面向对象分析过程	124
7.2.2	面向对象设计过程	126
7.2.3	设计模式	128
7.3	UML 概述	129
7.3.1	UML 发展历史	129
7.3.2	UML 简介	130
7.3.3	视图	131
7.3.4	图	133
7.4	小结	136
	习题	136
第 8 章	面向对象建模	137
8.1	用况建模	137
8.1.1	用况建模步骤	137
8.1.2	确定执行者	138
8.1.3	确定用况	139
8.1.4	用况描述	140
8.1.5	用况图中的关系	142
8.1.6	案例说明	143
8.1.7	用况建模实例	144

8.2 静态建模	147
8.2.1 类图和对象图	148
8.2.2 CRC 技术	148
8.2.3 类之间的关系	153
8.2.4 静态建模实例	161
8.3 动态建模	163
8.3.1 状态机图	163
8.3.2 活动图	170
8.3.3 顺序图	172
8.3.4 通信图	176
8.3.5 动态建模实例	178
8.4 物理体系结构建模	179
8.4.1 构件图	180
8.4.2 部署图	181
8.5 小结	182
习题	182
第 9 章 基于构件的软件开发	184
9.1 基于构件的软件开发概述	184
9.1.1 构件	184
9.1.2 基于构件的软件开发过程	187
9.1.3 CBSD 对质量、生产率和成本的影响	189
9.2 建造可复用构件	190
9.2.1 对可复用构件的要求	190
9.2.2 创建领域构件的设计框架	191
9.2.3 可变性分析	191
9.2.4 可变性机制	192
9.3 应用系统工程	194
9.3.1 基于 CBSD 的应用系统分析和设计	194
9.3.2 构件的鉴定、特化和组装	195
9.4 构件的管理	196
9.4.1 构件的分类描述	196
9.4.2 构件库管理系统	197
9.5 小结	198
习题	198
第 10 章 敏捷软件开发	199
10.1 敏捷软件开发方法概述	199
10.1.1 敏捷宣言	199

10.1.2	精益思想	201
10.1.3	敏捷方法综述	202
10.2	Scrum 方法	203
10.2.1	Scrum 简介	203
10.2.2	Scrum 团队	205
10.2.3	需求管理	205
10.2.4	基于时间盒的迭代	206
10.2.5	回顾会议	209
10.3	极限编程方法	210
10.3.1	极限编程简介	210
10.3.2	价值观和原则	211
10.3.3	实践	212
10.4	看板方法	217
10.4.1	看板方法简介	217
10.4.2	看板方法的规则	218
10.4.3	看板方法和 Scrum 的比较	219
10.5	小结	220
	习题	220
第 11 章	人机界面设计	221
11.1	人的因素	221
11.1.1	人对感知过程的认识	221
11.1.2	用户的技能和行为方式	222
11.1.3	人体测量学对设计的影响	222
11.2	人机界面风格	224
11.3	人机界面分析与建模	227
11.3.1	人机界面设计过程	227
11.3.2	人机界面设计中涉及的模型	228
11.3.3	任务分析的途径与方法	228
11.4	界面设计活动	229
11.4.1	定义界面对象和动作	229
11.4.2	设计问题	230
11.4.3	黄金原则	231
11.5	实现工具	233
11.6	设计评估	233
11.7	小结	235
	习题	235

第 12 章	程序设计语言和编码	236
12.1	程序设计语言	236
12.1.1	程序设计语言的基本成分	236
12.1.2	程序设计语言的特性	238
12.1.3	程序设计语言的分类	239
12.1.4	程序设计语言的选择	240
12.2	程序设计风格	241
12.2.1	源程序文档化	241
12.2.2	数据说明	244
12.2.3	语句结构	244
12.2.4	输入和输出	246
12.3	小结	246
习题		246
第 13 章	软件测试	247
13.1	软件测试基础	247
13.1.1	软件测试的目的	247
13.1.2	软件测试的基本原则	248
13.1.3	白盒测试和黑盒测试	249
13.2	白盒测试	249
13.2.1	逻辑覆盖测试	249
13.2.2	逻辑表达式错误敏感的测试	254
13.2.3	基本路径测试	255
13.2.4	数据流测试	257
13.2.5	循环测试	258
13.3	黑盒测试	259
13.3.1	等价类划分	259
13.3.2	边界值分析	262
13.3.3	比较测试	263
13.3.4	错误猜测	263
13.3.5	因果图	263
13.4	测试策略	267
13.4.1	V 模型	267
13.4.2	单元测试	268
13.4.3	集成测试	269
13.4.4	确认测试	272
13.4.5	系统测试	273
13.5	面向对象测试	274

13.5.1	面向对象语境对测试的影响	274
13.5.2	面向对象测试策略	275
13.5.3	面向对象测试用例设计	276
13.6	测试完成标准	276
13.7	调试	277
13.7.1	调试过程	277
13.7.2	调试方法	277
13.7.3	纠正错误	278
13.8	小结	279
	习题	279
第14章 Web工程		281
14.1	WebApp的属性和类型	281
14.1.1	WebApp的属性	281
14.1.2	WebApp的类型	283
14.2	Web工程过程	284
14.2.1	过程框架	284
14.2.2	改善框架	285
14.2.3	Web工程的最佳实践	285
14.2.4	方法和工具	285
14.3	WebApp建模	286
14.3.1	WebApp需求建模	286
14.3.2	WebApp设计建模	288
14.4	WebApp质量管理	293
14.4.1	WebApp质量维度	293
14.4.2	WebApp的测试	294
14.4.3	WebApp配置管理	295
14.5	小结	296
	习题	296
第15章 软件维护与再工程		297
15.1	软件维护	297
15.1.1	软件维护的概念	297
15.1.2	软件维护的过程	300
15.1.3	软件可维护性	302
15.2	再工程技术	305
15.2.1	再工程的概念	305
15.2.2	业务过程再工程	305
15.2.3	软件再工程过程	306

15.2.4 逆向工程	308
15.3 小结	309
习题	309
第 16 章 软件项目管理	310
16.1 软件项目管理概述	310
16.1.1 软件项目管理的关注点	310
16.1.2 软件项目管理的内容	313
16.2 软件度量	316
16.2.1 面向规模的度量	317
16.2.2 面向功能的度量	318
16.2.3 软件质量模型	321
16.2.4 程序复杂性度量	327
16.2.5 软件可靠性度量	329
16.3 软件项目估算	330
16.3.1 代码行、功能点和工作量估算	330
16.3.2 IBM 估算模型	331
16.3.3 CoCoMo 模型	332
16.3.4 Putnam 模型	334
16.3.5 软件可靠性估算	335
16.4 项目进度管理	336
16.4.1 人员与工作量之间的关系	337
16.4.2 任务的分解与并行	337
16.4.3 任务工作量的确定	338
16.4.4 进度安排	339
16.5 风险管理	344
16.5.1 风险标识	344
16.5.2 风险预测	345
16.5.3 风险评估	345
16.5.4 风险管理和监控	346
16.6 软件项目的组织	348
16.6.1 组织结构的模式	348
16.6.2 程序设计小组的组织形式	348
16.6.3 人员配备	350
16.7 软件质量管理	351
16.7.1 软件质量保证	352
16.7.2 软件评审	353
16.8 软件配置管理	355
16.8.1 软件配置管理的基本概念	355

16.8.2 软件配置管理的主要活动.....	356
16.9 小结.....	359
习题.....	359
词汇索引	361
参考文献	371

第 1 章

概论

自从 1946 年第一台数字电子计算机问世以来,计算机硬件经历了电子管、晶体管、集成电路、大规模集成电路等多个时代,得到了飞速的发展。计算机的性能越来越高,而价格越来越便宜。计算机的应用也从单纯的科学计算,很快渗透到工业、农业、国防、商业等各个领域。随着个人计算机的诞生,办公自动化和管理信息系统的深入普及,特别是网络技术的飞速发展,计算机已进入家庭,人们可以通过计算机和网络,方便地进行信息的获取、处理和交流。电子政务、电子商务不断地涌现,使计算机成为人们工作和生活不可缺少的工具,同时也改变了传统的办事模式,计算机基础知识与技能也已成为新时代人们的基本文化素养。

随着计算机应用的深入,对计算机软件需求量越来越大,对软件的功能性、易使用性、可靠性等要求也越来越高。为了在有限的资金、资源和时间条件下开发满足客户要求的高质量软件,就需要研究与软件开发和管理相关的模型、方法、技术、过程、工具和环境等,这些就是软件工程研究的主要内容。

本章介绍计算机软件和软件工程的基本概念,包括计算机软件、软件工程、软件过程、软件过程模型、CASE 工具和环境。

1.1 计算机软件

在《计算机科学技术百科全书》中,对计算机软件作出如下定义:计算机软件指计算机系统程序及其文档。程序是计算任务的处理对象和处理规则的描述。任何以计算机为处理工具的任务都是计算任务。处理对象是数据(如数字、文字、图形、图像、声音等,它们只是表示,而无含义)或信息(数据及有关的含义)。处理规则一般指处理的动作和步骤。文档是为了便于了解程序所需的阐述性资料。

1.1.1 软件的发展

自第一台计算机问世以来,计算机软件经历了多个发展阶段。关于软件的发展阶段有多种说法,徐家福教授在《计算机科学技术百科全书》中将软件的发展分为如下 3 个阶段。

第一阶段(1946—1956 年):从第一台计算机上的第一个程序的出现到实用的高级程序设计语言出现以前。在这个阶段,计算机的存储容量比较小,运算速度比较慢;编写程序

的工具只有低级语言,即以机器基本指令集为主的机器语言和在机器语言基础上稍加符号化的汇编语言;计算机的应用领域主要是以数值数据处理为主的科学计算,其特点是输入输出量较小,但计算量却较大。由于计算机的容量小、速度慢,又采用低级语言编程,所以程序的设计和编制工作复杂、繁琐、费时且易出差错,强调编程技巧,很少考虑程序结构的清晰性、易读性和易维护性。衡量程序质量的标准主要是功效,即运行时间省、占用内存小。由于当时的程序规模都比较小,所以设计和编制程序主要采用个体工作方式。当时尚未出现“软件”一词,人们对与程序有关的文档的重要性认识不足,开发的软件除了程序外,几乎没有其他的文档。当时的主要研究内容是科学计算程序、服务性程序和程序库,研究对象是顺序程序。

第二阶段(1956—1968年):从实用的高级程序设计语言出现以后到软件工程出现以前。在这个阶段,随着计算机硬件的飞速发展,出现了大容量的存储器,外围设备也得到了迅速发展;出现了高级程序设计语言,使得编制程序的工作从专业人员扩展到工程技术人员;计算机的应用领域逐步扩大,出现了大量的数据处理问题,其性质和科学计算有明显的区别,涉及非数值数据,其特点是计算量不大,但输入输出量却较大。计算机硬件的飞速发展使得高速主机与低速外围设备的矛盾日益突出,为了充分利用系统资源,出现了操作系统。为了适应大量数据处理问题的需要,开始出现数据库及其管理系统。在20世纪50年代后期,人们逐渐认识到文档的重要性,到20世纪60年代初期,出现了融程序及其有关文档为一体的“软件”一词。这一阶段的研究对象增加了并发程序,着重研究高级程序设计语言、编译程序、操作系统及各种应用软件,研究出指导和辅助编程的一些方法和工具,如结构化程序设计方法,排错工具等。由于软件规模的日益增大,设计与编制程序的工作方式逐步从个体方式转向合作方式。随着计算机硬件的发展和高级程序设计语言的出现,计算机应用得到迅速发展。由于缺乏有效的工程化方法的指导,使得很多软件不能按计划完成,有的甚至夭折,大量已有软件难以维护,到20世纪60年代中期,出现了人们难以控制的局面,即所谓的软件危机,从而导致软件工程的出现。

第三阶段(1968年以来):从软件工程出现以后至今。在这个阶段,计算机硬件向巨型机和微型机两个方向发展,出现了计算机网络,特别是Internet得到了飞速的发展;软件方面提出了软件工程,用工程化方法管理和开发软件;计算机的应用领域渗透到各个业务领域,出现了嵌入式应用,其特点是受制于所嵌入的宿主系统,而不只是受制于其功能要求。随着微型计算机和计算机网络的发展,网络软件、分布式应用和分布式软件得到发展。开发方式逐步由个体合作方式转向工程方式,软件工程发展迅速,出现了“计算机辅助软件工程”(computer aided software engineering, CASE)。软件工程方面的研究主要包括软件开发模型、软件开发方法及技术、软件工具与环境、软件过程、软件自动化系统等。除了软件传统技术继续发展外,人们还着重研究以智能化、自动化、集成化、并行化以及自然化为标志的软件开发新技术。

1.1.2 软件的特点

软件与硬件相比有以下不同的特点:

- 软件是一种逻辑实体,而不是有形的系统元件,其开发成本和进度难以准确估算。

- 软件是被开发的或被设计的,没有明显的制造过程,一旦开发成功,只需复制即可,但其维护的工作量大。
- 软件的使用没有硬件那样的机械磨损和老化问题。硬件在使用的初期由于设计或制造上的问题,可能有较高的故障率,故障一旦修复以后,故障率会降到一个稳定的水平上,之后随着机械磨损、老化等因素,故障率又会急剧上升。硬件的故障曲线如图 1.1 所示。而软件在交付使用前通常已经过严格的测试,排除了已发现的错误,在使用初期,那些未发现的错误会引起较高的故障率,由于软件没有磨损、老化等问题,故这些故障一旦修复,软件的故障曲线就为如图 1.2 所示的理想曲线。然而,在软件使用过程中都有维护问题,软件的维护需要修改程序,修改可能引入副作用,从而使故障率升高。因此,软件的实际故障曲线如图 1.2 所示^[2]。

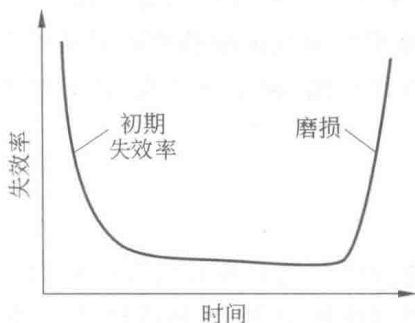


图 1.1 硬件的故障曲线

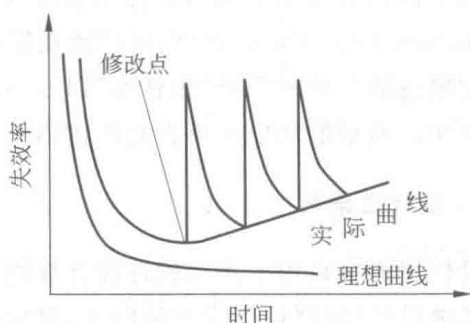


图 1.2 软件的故障曲线

1.1.3 软件的分类型

在《计算机科学技术百科全书》中,将软件分为系统软件、支撑软件和应用软件 3 类。

1. 系统软件

系统软件居于计算机系统中最靠近硬件的一层,其他软件一般都通过系统软件发挥作用。系统软件与具体的应用领域无关,例如,编译程序、操作系统等。

2. 支撑软件

支撑软件是支撑软件的开发、维护与运行的软件。例如,数据库管理系统、网络软件、软件工具、软件开发环境等。

3. 应用软件

应用软件是特定应用领域专用的软件。例如,人口普查的软件、嵌入式应用软件等。

1.1.4 软件语言

编制程序离不开程序设计语言,书写计算机软件就可能使用软件语言。软件语言是用于书写计算机软件的语言,主要包括:需求定义语言、功能性语言、设计性语言、程序设计语

言和文档语言。

1. 需求定义语言

需求定义语言是用于书写软件需求定义的语言。软件需求包括功能需求和非功能需求两个方面。功能需求从用户角度明确了软件系统必须具有的功能行为,它是整个软件需求的核心所在。在功能需求的基础上,非功能需求对软件需求作进一步的刻画,包括功能限制、设计限制、环境描述、数据与通信规程和项目管理等。典型的需求定义语言有 PSL (problem statement language)等。

2. 功能性语言

功能性语言是用于书写软件功能规约的语言,通常又称为功能规约语言。软件功能规约(functional specification)也称功能规格说明,是软件所要完成功能的精确而完整的陈述,通常它描述的是软件系统“做什么”以及“只做什么”的外部功能,而不涉及系统“如何做”的内部算法。典型的功能性语言有广谱语言、Z语言等。

3. 设计性语言

设计性语言是用于书写软件设计规约的语言。软件设计规约是软件设计(包括总体设计和详细设计)的严格而完整的陈述,是软件功能规约的算法性细化,既刻画软件“如何做”的内部算法,同时又是软件实现的依据。典型的设计性语言有 PDL (program design language)。

4. 程序设计语言

程序设计语言(programming language)是用于书写计算机程序(习惯上指实现级语言程序)的语言。计算机程序是计算任务的处理对象和处理规则的描述。

程序设计语言可以按照语言的级别、对用户的要求、应用范围、使用方式、成分性质等多种角度进行分类。

(1) 按语言级别可分为低级语言和高级语言

低级语言是与特定计算机体系结构密切相关的程序设计语言,如机器语言、汇编语言。其特点是与机器有关,功效高,但使用复杂、繁琐、费时、易出差错,难以维护。

高级语言是不反映特定计算机体系结构的程序设计语言,它的表示方法比低级语言更接近于待解问题的表示方法。其特点是在一定程度上与具体机器无关,易学、易用、易维护。但高级语言程序经编译后产生的目标程序往往功效较低。

(2) 按用户要求可分为过程式语言和非过程式语言

过程式语言(procedural language)是通过指明一系列可执行的运算及运算次序来描述计算过程的程序设计语言,如 FORTRAN、C、Java 等。

非过程式语言(nonprocedural language)是不显式指明处理过程细节的程序设计语言。在这种语言中尽量引进各种抽象度较高的非过程性描述手段,以期做到在程序中增加“做什么”的描述成分,减少“如何做”的细节描述,如 PROLOG 等。