EECS

电子工程与计算机科学

面向对象软件工程实践指南

曹 健 编著

上海交通大学出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

面向对象软件工程实践指南

EECS

电子工程与计算机科学

曹 健 编著

上海交通大學出版社
SHANGHAI JIAO TONG UNIVERSITY PRESS

内容提要

本书围绕基于面向对象方法学的软件开发过程，介绍了各个典型环节和各个环节中采用的技术，并给出了一个详细完整的案例。主要内容为：面向对象软件工程基本概念和统一建模语言 UML 的介绍，在此基础上，对软件开发计划、需求定义、分析、设计、构造、测试、交付和总结等各个阶段的步骤、采用的技术和交付物进行了阐述。书中给出了一个详细的案例，与每一个环节相对应。读者可以通过学习前半部分的指南并参考后半部分的案例了解软件开发过程的组织和实施的具体方式。

本书可以作为高等院校计算机科学与技术、软件工程以及其他相关学科的软件工程课程的配套教材，也可供研究生、工程技术人员进行参考。

图书在版编目(CIP)数据

面向对象软件工程实践指南 / 曹健编著. —上海：
上海交通大学出版社，2017
ISBN 978-7-313-16218-2

Ⅰ. ①面… Ⅱ. ①曹… Ⅲ. ①面向对象语言—程序设计 Ⅳ. ①TP312.8

中国版本图书馆 CIP 数据核字(2016)第 288268 号

面向对象软件工程实践指南

编　　著：曹　健
出版发行：上海交通大学出版社
邮政编码：200030
出 版 人：郑益慧
印　　制：常熟市文化印刷有限公司
开　　本：787 mm×1092 mm　1/16
字　　数：432 千字
版　　次：2017 年 2 月第 1 版
书　　号：ISBN 978-7-313-16218-2/TP
定　　价：48.00 元

地　　址：上海市番禺路 951 号
电　　话：021-64071208
经　　销：全国新华书店
印　　张：17.75
印　　次：2017 年 2 月第 1 次印刷

前言

软件的广泛使用已经成为驱动社会发展的重要力量。“软件工程”作为一门研究系统、规范、合理化软件开发的学科，是计算机专业、软件工程专业的核心课程，也是其他专业可能选修的课程。目前已经有许多的软件工程教科书，不少教科书还是这个领域的经典。显然，这些教科书在解释软件工程的相关理论、技术方面均有自己独特之处。然而，软件工程不是一门单纯理论性的课程，学生在学习时除了完成一些作业外，必须能够进行软件工程的实践，而且这种实践不是个别环节、个别技术的。因此，以小组为单位让学生能够以软件工程方法为指导完整地开发一个软件系统就非常有必要了。目前，许多学校在教这一门课程时，也确实要求学生进行小组项目的开发。然而，在这个过程中，软件工程教科书由于以讲解知识点为主，往往并未给以明确的实践性指导。

笔者教授软件工程多年，尝试过不同形式的教学方式，深感到一本软件工程实践教程的必要性。虽然国内也有一些软件工程实践方面的教材，笔者还是觉得指导性不够，学生还是难以一步一步“按图索骥”地完成一个软件项目的完整训练。因此，笔者在总结多年教学经验的基础上，针对面向对象软件工程来提供软件工程项目的训练教程。在实际中，软件项目的组织方式包括软件过程模型、文档模板，涉及的模型、采用的方法是多样化的。本书的目的主要在于为学生进行软件项目实践提供指导，所以选择了传统的开发过程模型，并对文档模板和模型集合进行了挑选。在具体的内容组织上，提供了完整的实际的案例，以给予学生直观的参考。当然，本书自身的知识体系也是完整的，对面向对象软件工程有兴趣的读者可以通过本书了解相关的知识。

本书的编写得到了俞嘉地、盛斌、薛庆水老师的协助，也得到了研究生姚艳、贾挺杰、刘辰旸、顾颀、李键、马子泰、朱能军、刘涛的帮助。本书中的案例编写来源于雷浩若、徐

源、田晓亮、姚佳乐、苏畅同学的实际作业，并经过了田晓亮同学和笔者的进一步修改。在此，对所有帮助过本书编写的人一并表示感谢。

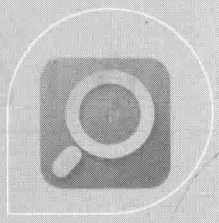
由于笔者水平有限，书中可能会出现表达不够准确的地方，敬请读者指正。书中的案例也仅仅出于示范的目的而提供，并非最佳设计，甚至可能存在缺陷，也请读者在参考时加以注意。

希望通过本书，能够促进软件工程教学效果的提升。

目 录

第一篇 指南篇

第二篇　案 例 篇



第一篇
指南篇

第1章　软件工程概论

软件工程是在软件的应用得到逐步推广的过程中自然产生的。一方面，它逐渐发展出了针对软件开发所遇到的共性问题的系列技术和方法，另一方面，它又遵从并不断借鉴了人类在其他领域发展而来的工程化的思想。

通过回顾软件工程的历史，并了解其运用的工程思想，特别是目前流行的面向对象的软件工程思想，有助于我们理解掌握软件工程方法。

1.1　软件工程的发展历史

1.1.1　第一台计算机和第一位程序员

世界上公认的第一台电子计算机是 ENIAC(埃尼亚克)，它问世于 1946 年 2 月 14 日，全称是“电子数值积分计算机”，英文名为“Electronic Numerical Integrator and Computer”，它是由美国宾夕法尼亚大学莫尔电子工程学院的莫尔小组承担研制的(见图 1-1)。

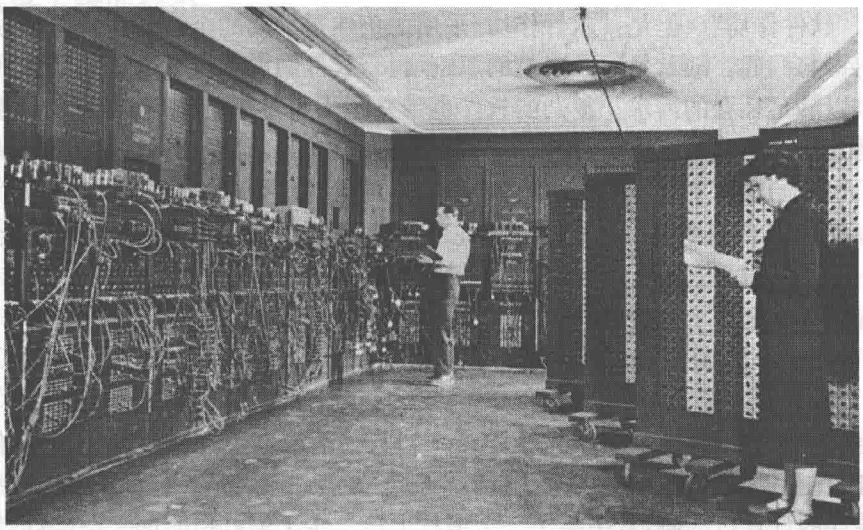


图 1-1　世界上第一台电子计算机 ENIAC

但是，世界上首位程序员的出现却远远早于第一台电子计算机，并且这个第一位程序员是位女士。她的名字叫Augusta Ada LoveLace，1815年生于伦敦(见图1-2)。Ada设计了巴贝奇分析机上求解伯努利方程的一个程序，并证明了巴贝奇的分析器可以求解许多问题。在1843年Ada发表的一篇论文里面提出机器可以用来创作音乐、制图以及进行科学研究。同时，Ada还提出了循环和子程序等计算机的重要概念，为计算设计了“算法”，并创作出了“程序设计流程图”。因此，Ada被广泛地认为是世界上第一位程序员。为了纪念她，1980年12月10日，一种新的计算机编程语言以她的名字命名，那就是Ada。Ada曾广泛用于美国军方尖端武器开发中。

图1-2 世界上第一位程序员
Augusta Ada LoveLace

1.1.2 软件的发展和软件危机

20世纪50年代，伴随着第一台电子计算机的问世，编程语言开始出现，相应地，计算机软件诞生了，以写软件为职业的人也逐渐开始出现，他们是真正意义上的程序员。最开始，这些程序员大多是一些有经验和经过训练的电子工程师甚至是数学家。

在计算机发展的初期(20世纪60年代中期以前)，计算机主要用于军事领域，后来才慢慢普及到民用领域。那时候硬件作用十分单一，通常只能用来执行一个程序。当时的计算机硬件也非常昂贵，编程人员必须在有限的处理器性能和极小的储存空间限制下，编写出执行速度快、占用空间小的程序，因而程序的编写充满了各种技巧，又带有个人色彩。当时的软件开发主要依赖程序员的聪明才智，同时，软件除了源代码以外，几乎没有文档等附属产品。软件的开发没有什么系统的方法。

20世纪60年代到70年代，计算机领域进入了比较快速的发展时期，正是在这一时期，软件从军用扩展到了民用，并作为一种广泛存在的产品为人们所接受。这个时期的一个重要特征是出现了“软件作坊”。但是，“软件作坊”采用的仍然是早期的个体软件开发的方式，几乎没有团队的协调与沟通。随着软件需求量的急剧增长，软件的需求也日益复杂，个体化开发的方式越来越难以满足社会的需求。复杂的软件也带来了大量的维护问题，然而许多程序的个体化特征使得它们最终成为不可维护的。随着计算机应用的日益普及，软件数量急剧上升，失败的软件项目也开始层出不穷，这一现象引起了普遍的关注，因而出现了“软件危机”这一说法。1968年北大西洋公约组织的计算机科学家在联邦德国召开学术会议，正是在这次会议上，“软件危机”第一次正式提出。

下面介绍5起历史上著名的软件灾难：

1) 水手号(Mariner)的致命BUG(1962年)

损失：1 850万美元

携带空间探测器的水手1号(The Mariner 1)火箭前往金星，在起飞后不久就偏离了预定航线。任务控制系统不得不在起飞293秒后摧毁了火箭。原因是一名程序员把一条手写的公式编写为错误的计算机代码，其中漏了一个横杠上标。少了横杠指明的平滑函数，软件就把速率的正规变分视为严重的错误，并对该错误进行了修正，从而将火箭引导偏离了航向。

2）哈特福德体育场倒塌事件(1978 年)

损失：7 000 万美元，以及给当地经济造成的 2 000 万美元损失

当成千上万的球迷离开哈特福德体育场几个小时后，钢结构的体育场屋顶就被湿雪压垮了。原因是 CAD 软件的程序员在设计体育场时错误地假设钢结构屋顶的支撑仅承受纯压力。但当其中的一个支撑意外地因大雪垮塌后，引发了连锁反应，导致屋顶的其余部分像多米诺骨牌一样相继倒掉。

3）苏联天然气管道爆炸(1982 年)

损失：数百万美元，并严重破坏了苏联经济

控制软件出故障造成跨西伯利亚输气管道压力急剧上升，导致了历史上最大的人为非核爆炸。据说，CIA 侦探在苏联购买的用于控制输气管道的系统内植入了一个 BUG。

4）几乎引发第三次世界大战的导弹误报事件(1983 年)

损失：将近全人类的毁灭

苏联预警系统误报美国发射了 5 枚弹道导弹。幸运的是，苏联的执勤官认为如果美国真的要攻击苏联的话，发射的导弹肯定不止 5 枚，因此他把这次攻击报告界定为一次误报。误报的原因是苏联预警系统中有一个 BUG，该系统误将阳光反射云顶识别为导弹。

5）医疗器械致死案(1985 年)

损失：死亡 3 人，严重受伤 3 人

加拿大的 Therac－25 放射治疗仪发生了故障，令病人受到了致命的辐射。原因是软件中一个称为竞态条件(race condition)的细小 BUG，一名技术人员可能在病人尚未进行适当防护的情况下意外地将 Therac－25 配置为高能模式。

在 http://www.devtopics.com/20－famous-software-disasters/上可以找到更多这样的例子。软件危机的出现，让人们对软件的开发有了更深入的研究和更多的反思，并开始改变对软件的一些不正确看法。易懂、易用、易修改、易维护等软件工程提倡的理念逐渐被大众所接受。

1.1.3 软件工程的提出

在 1968 年北大西洋公约组织的计算机科学家的会议上集中讨论了如何应对“软件危机”，在这次会议上，第一次提出了“软件工程”。

“软件工程”是一门研究系统、规范、合理化软件开发的学科。软件工程运用工程学的原则和方法重新制订了软件开发的流程和方案。具体来说，软件工程涉及两大方面主要内容，首先是软件开发的技术，其次是软件开发的管理。这二者缺一不可。其中软件开发技术主要包括了软件开发方法、工具、环境等，软件开发管理则包括了软件开发周期管理、开发人员管理、进度管理等内容。

软件工程发展至今，大致可以分为结构化软件工程(也称为传统软件工程)和面向对象软件工程(也称为现代软件工程)。结构化软件工程围绕功能、数据和数据流展开分析和设计，以模块为中心，自顶向下、逐步求精完成软件设计，系统是实现模块功能的函数和过程的集合。而面向对象软件工程则以对象为核心，通过识别系统中的类，定义对象之间的交互，考虑类的代码实现从而完成系统分析和设计。

然而，软件工程目前依然不够成熟。不同的人对软件开发持有不同的观点，如以 C. A. R.

Hore 为代表的数学观，以 Bertrand Meyer 为代表的工程观，以 Ivar Jacobson 为代表的建模观等。而在现实生活中，许多程序员还认为软件开发是个手工艺活，还有一些人甚至把软件开发看作是一门“艺术”——不同的人发挥自己的创造力写出迥然不同的代码。因而，要使得软件开发逐步成熟，要大力传播软件工程的思想，同时软件工程自身还需要不断发展完善。

1.2 软件工程基本思想

无论是传统软件工程还是面向对象软件工程，它们都体现了一些共同的思想，这些思想主要有：抽象，分解，分类，复用。

1.2.1 抽象

抽象，是人类解决复杂问题的通用方法。抽象是从众多的事物中抽取出共同的、本质性的特征，而舍弃其非本质的特征。通过硬件基础上运行的软件来解决实际问题时，软件中的概念和实际问题中的概念是有区别的，因此必须采用抽象来实现实际问题在软件世界中的映射。在传统软件工程中，问题被映射成函数、数据结构、算法等软件概念，而在面向对象软件工程中，问题被映射成对象、类以及它们之间的关系，由于对象、类模拟了现实世界，这种抽象更容易理解。为了实现从问题领域到软件领域的映射，软件工程把软件开发分成了多个阶段，每一个阶段中提供了多种模型来完成任务，而模型本身就是一种抽象表达。

1.2.2 分解

分解，也是人类解决复杂问题的通用方法。所谓分解，就是把复杂的系统变成小的系统，采用“各个击破”的原则逐一解决。由于软件本身比较复杂，作为一个整体开发存在一定困难，因此，把软件系统分解成一个个小系统，这样就可以大大降低开发难度。传统的软件工程在分解时，从功能角度出发，各个子系统都对应了一部分功能；而面向对象的软件工程中，把系统分解为一个个对象，通过定义对象间的交互来完成所有的功能。分解也促进了软件重用，由于每一个小的单元（子系统、模块、类、函数）具备一定的功能，在未来的软件开发中可以再次使用，那些具有一定通用性的软件，甚至可以构成一个可重用软件库。

1.2.3 复用

复用，就是利用已有的代码，或者已有的知识、经验编写代码，以进行新的软件开发。复用可以节省很大一部分时间和精力，从而提高开发效率。复用的软件大多经过很长时间的检验，这样可以减少开发过程中可能出现的错误。小部分的创新加上大部分的已有成果来完成新项目，因此利用复用可以高效而又高质量地完成软件开发工作。

复用的形式有多种多样，主要的形式为程序库、类库、软件服务、应用框架、设计模式等。

(1) 程序库是代码复用最直接的例子。有很多的函数或者模块在软件之间都是通用的，比如说对一个数组的排序函数，文件的读、写函数等。将这些函数、模块封装在一个程序库中，后续进行开发工作的程序员只需要简单地进行调用就可以了。程序库的好处是显而易见的：程序员通过调用相应函数或模块，可以提高开发的效率；同时，程序库中的操作一定是经过充