HZ BOOKS
华章教育

面向CS2013计算机专业规划教材

软件工程概论

第2版

郑人杰 马素霞 殷人昆 编著

An Introduction to Software Engineering

Second Edition

机械工业出版社
China Machine Press

面向CS2013计算机专业规划教材

软件工程概论

第2版

郑人杰 马素霞 殷人昆 编著

An Introduction to Software Engineering

Second Edition

机械工业出版社
China Machine Press

图书在版编目（CIP）数据

软件工程概论 / 郑人杰，马素霞，殷人昆编著 . —2 版 . —北京：机械工业出版社，2014.9
（面向 CS2013 计算机专业规划教材）

ISBN 978-7-111-47821-8

I. 软… II. ① 郑… ② 马… ③ 殷… 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（2014）第 204142 号

软件工程学科具有知识面广、发展迅速、实践性强等特点。本书作者针对软件工程的学科特点，在写作中注重结合实例讲解软件工程的理论与方法，避免抽象和枯燥的论述，在兼顾传统的结构化方法的同时，注重当前广为采用的面向对象方法。全书内容组织成五部分：第一部分是软件工程概述；第二部分介绍结构化分析与设计方法；第三部分讲述面向对象分析与设计方法；第四部分讲解软件实现与测试；第五部分介绍软件维护与软件管理。

本书结构合理、内容丰富，讲解通俗易懂、由浅入深，适合作为计算机科学与技术、软件工程等专业本科生的教材。

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：刘立卿　　责任校对：殷　虹
印　　刷：北京诚信伟业印刷有限公司　　版　　次：2014 年 11 月第 2 版第 1 次印刷
开　　本：185mm × 260mm　1/16　　印　　张：23.75
书　　号：ISBN 978-7-111-47821-8　　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

前　言

当今，软件业是社会经济发展的先导性和战略性产业，它已成为信息产业和国民经济新的增长点和重要支柱。软件工程在软件开发中起着重要的作用，对软件产业的形成及发展起着决定性的推动作用。采用先进的工程化方法进行软件开发和生产是实现软件产业化的关键技术手段。与其他产业相比，软件产业具有自己的特殊性。软件产业的发展更加依赖于人力资源，因此软件产业的竞争越来越集中到对人才的竞争。然而，刚毕业的大学生往往要经过半年到一年的培训才能适应软件企业的工作。长期以来，我国软件人才的现状远远不能满足软件产业发展的要求。因此，软件工程人员队伍的成长，特别是高层软件工程人员队伍的成长显得更为紧迫。

自从软件工程概念诞生以来，学术界和工业界做了大量的研究与实践工作，也取得了许多重要成果。尤其是上个世纪 90 年代以后，随着网络技术及面向对象技术的广泛应用，软件工程取得了突飞猛进的发展。软件工程已从计算机科学与技术中脱离出来，逐渐形成了一门独立的学科。软件工程教育所处的地位也越来越重要，软件工程课程已成为软件工程、计算机科学与技术等专业的必修课程。

软件工程课程实践性比较强，如果学生没有实践经验，则很难理解相关的理论知识。因此，教师普遍感到软件工程课程难教，而学生则普遍感到难学。近年来，软件工程学科的发展非常迅速，新的理论、方法和工具层出不穷，其中很多已经应用到企业的实际工作中。软件工程的教学面临越来越大的压力。我们认为，除了需要在教学内容、教学方法方面进行改革之外，实践能力的培养对于建设一支企业需要的合格软件工程人才队伍显得更为关键。

我们在编写中力图遵循如下原则：

(1) 既要强调和突出基本概念、基本方法，又要尽可能使材料内容的组织符合学生的认识规律，在讲解理论的过程中尽量结合实例，并注重软件工程方法、技术和工具的综合应用，避免讲解抽象和枯燥。

(2) 在兼顾传统的结构化方法的同时，注重当前广为采用的面向对象方法。紧密结合当前技术的新发展，在阐述理论知识的同时侧重实用性。

(3) 既要充分重视技术性内容，使其作为初学者必须掌握的知识，同时也要兼顾软件工程实践中必不可少的管理知识。

本书在第 1 版的基础上对部分内容作了调整和充实。例如第 1 章增加了软件工程工具的介绍和软件工程方法概述；同时将软件生存期模型抽出来专设了第 2 章，并且增加了“敏捷过程”等。此外也更新了一些内容，如第 1 章的软件工程知识体及知识域、第 12 章的 CMMI 和第 14 章的软件工程标准等。

总之，本书力争做到结构合理、内容丰富，讲解由浅入深，既体现知识点的连贯性、完整性，又体现其在实际中的应用。

教学建议

第1章　软件与软件工程的概念（3～4学时）

本章主要是概念性和基础性的内容，包括软件的概念、特性及软件危机的主要表现，软件工程的概念及软件生存期。对软件工程方法进行了简要介绍，包括传统的结构化方法、面向对象方法及形式化方法。鉴于软件工具是软件工程的三个要素之一，而且在软件开发中起着越来越重要的作用，本章对软件工具进行了介绍。

最后一节是软件工程知识体系及知识域的介绍，它把软件工程的全景图展现给读者，属于了解性的内容，可以让学生课后阅读相关的参考文献。

第2章　软件生存期模型（3～4学时）

本章介绍应用比较广泛的传统软件生存期模型及现代软件生存期模型。传统软件生存期模型包括瀑布模型、快速原型模型、增量模型、螺旋模型、喷泉模型等；现代软件生存期模型包括统一过程、基于构件的开发模型、敏捷过程等。

第3章　软件需求获取与结构化分析方法（4～6学时）

3.1节和3.2节需要详细讲解，同时也要求学生较好地准确掌握。3.1节的内容不仅适用于传统方法，而且也适用于面向对象方法。3.2节对数据流分析方法做了较详细的介绍，画分层的数据流图、E-R图及状态图是本节的重点，书中结合实例进行了讲解。学生在课后应该进行实践才能逐渐掌握上述知识。在实践中最容易出现的问题是数据流的分层，由抽象到具体的程度难以把握。很多同学一开始就画出非常详细的数据流图，往往图太复杂，看不清楚，失去了建模的意义。如果层次难以把握，最起码应强调先画出顶层数据流图，正确的顶层数据流图是一个好的开端，也是设计阶段中外部接口设计和交互设计的依据。

第4章　结构化设计方法（5～6学时）

在传统方法中，体系结构设计实质上就是软件结构设计或模块结构设计。4.3节讲解了基于数据流的设计方法，重点讲解了如何使用变换型映射方法和事务型映射方法生成初始模块结构，以及如何对模块结构进行改进，最后给出了一个设计与改进的例子。这部分内容需要重点讲解。

对于大多数应用系统来说，人机交互（用户）界面设计是接口设计的主要内容。人机交互界面的设计越来越受到重视，因为很多使用方面的特性都体现在用户界面上，用户往往会根据界面来评价软件。这方面是最难于把握的，因为不同的人会有不同的审美观点和不同的使用习惯。4.4节介绍了界面设计类型及人机交互的设计准则。

4.5节介绍了文件设计和数据库设计。在数据库设计部分主要讲解了如何将E-R图映

射到数据库中的表。考虑到数据库的具体设计在数据库原理与应用等课程中学过，因此本节没有再展开。

4.6 节是需要重点讲解和实践的内容。老师可能会认为算法的描述在程序设计类课程、数据结构与算法课程中都学习和实践过，在软件工程课程中不需要重点介绍。但在实际项目开发中，大多数学生在这方面都表现不佳。算法图中的错误很多，很多情况下都不是结构化的，往往要经过多次修改才能符合要求。自顶向下、逐步细化的方法也运用得不好，致使在描述复杂算法时很困难。

第 5 章　面向对象方法与 UML（4 学时）

本章对面向对象概念与开发方法进行了介绍，对 UML 的基本模型、事物、关系及建模时用到的各种图进行了介绍。由于采用面向对象方法建模普遍使用 UML 来描述，因此本章需要重点讲解。虽然后面在面向对象分析、面向对象设计中还会涉及这些内容，但在那里基本上都是使用，细节的地方没有再讲解。

在组织教学时教师可以根据需要对顺序进行调整，将有些部分移到后面的章节，与分析和设计结合起来讲解。

第 6 章　面向对象分析（4 学时）

本章主要讲解面向对象分析的 3 种模型，即功能模型（用例模型）、静态模型（对象模型）和动态模型（交互模型）。在讲解中结合实例，避免了理论讲解太抽象和枯燥。

本章的内容需要重点讲解，学生需要在课后进行实践。

第 7 章　软件体系结构与设计模式（4～6 学时）

本章内容比较抽象，大多数本科生接受起来比较困难，教师可以根据具体情况有选择地进行讲解，如认为不必要或学时有限也可以不讲。

第 8 章　面向对象设计（4～6 学时）

面向对象设计是面向对象分析的继续，也是面向对象分析的深入和细化，重点是对面向对象分析时建立的对象模型进行调整和细化，另外需要考虑与实现有关的方面，如数据存储设计、人机交互界面设计等。

从总体上来说，与传统的结构化方法相比，面向对象设计与面向对象分析具有更好的连续性，在方法和工具的使用上保持了高度的一致性。

第 9 章　软件实现（3～4 学时）

本章重点讲解程序设计风格和编码规范。大多数初学者和软件工作新手喜欢学习编码技术，但不愿意花时间写注释，很多方面喜欢随心所欲，不愿意受规范的束缚。主要原因是他们并没有认识到风格和规范的重要性，认为实现了功能就万事大吉了。对于学生完成作业和练习来说，这种想法无伤大雅，因为别人不需要看，以后也不需要再修改了。但对于企业开发的系统来说，完成了功能才只是开始，以后面临的是漫长和繁重的维护任务。况且现代软件开发都是团队开发，如何使很多人开发的代码就如同一个人开发的一样，使

得互相都能读懂且易于阅读，解决的办法就是要遵循统一的规范。

第 10 章　软件测试方法（6 学时）

本章需要学生了解与软件测试相关的概念、软件测试的重要性及软件测试与开发各个阶段的关系，同时也要了解穷举测试是不可能的，测试只能证明软件存在错误，而不能证明软件中不存在错误。白盒测试技术及黑盒测试技术是需要重点讲解的内容，也是读者必须掌握的内容。

第 11 章　软件维护（2～4 学时）

本章对软件维护的任务、维护活动及维护方法进行了介绍。要求学生了解为什么要对软件进行维护，有哪些种类的维护以及软件维护的重要性。维护不仅仅是修改程序，也需要遵循一定的过程。这章内容也可由教师根据教学安排的实际情况做裁剪和取舍。

第 12 章　软件过程与软件过程改进（2～3 学时）

本章介绍了软件过程的定义、分类以及软件过程改进模型 CMM/CMMI。该章属于了解性内容，如果学时有限，可以有选择地介绍。

第 13 章　软件项目管理（4～6 学时）

本章重点讲解软件生产率和质量的度量、软件项目的时间管理、软件项目的成本管理及软件配置管理。人员管理和风险管理部分可以让学生课后阅读。

第 14 章　软件工程标准及软件文档（2 学时）

软件工程的标准化工作对于推动软件的产业化具有重要意义，而文档是软件不可分割的一部分，对软件的维护起着重要作用。本章要求学生了解软件工程标准的分类和意义，从而增强软件工程项目的标准化意识，为今后工作中的应用打下基础。此外，通过这章的学习，读者还应了解文档的作用与分类以及文档的管理与维护。

目　录

第四部分　软件实现与测试

第五部分　软件维护与软件管理

第一部分
PART ONE

软件工程概述

CHAPTER 1

第 1 章

软件与软件工程的概念

计算机技术经过了 60 多年的发展历程，取得了突飞猛进的发展。计算机的应用领域从单纯的科学计算发展到军事、经济、教育、文化等社会生产及生活的各个方面，推动了其他行业及领域的发展，改变了人们学习、工作及生活的方式。

计算机软件系统是信息化的重要组成部分。计算机软件已形成了独立的产业，成为国民经济新的增长点和重要支柱。软件工程在软件开发中起着重要的作用，对软件产业的形成及发展起着决定性的推动作用。

进入 21 世纪，人类已从工业社会跨入了信息社会。软件工程本身也取得了突飞猛进的发展，逐渐从计算机科学与技术学科中分离出来，已经形成了比较完整的软件工程学科体系。

本章将对软件工程相关的概念、软件生存期、软件开发方法及工具进行简要介绍，使读者对软件工程的总体框架获得初步的了解。

1.1 软件的概念、特性和分类

1.1.1 软件的概念及特性

1. 软件的概念

我们国家 20 世纪 80 年代初的大学生知道软件的人并不多，甚至很多人从未听说过这个词，即使是当初软件专业毕业的学生也不曾想到软件的发展速度如此之快。今天的软件已无处不在，渗透到了各个行业之中，并在很大程度上提高了各个行业的自动化水平。随着计算机大量进入家庭，计算机已经成为我们日常生活、学习和工作都离不开的工具。

虽然软件对于现代人来说并不陌生，但很多人对于软件的理解并不准确，“软件就是程序，软件开发就是编程序”这种错误观点仍然存在。因此，仍然有必要给软件一个明确的定义。

软件的一种公认的传统定义为：软件是计算机系统中与硬件相互依存的另一部分，包括程序、数据及其相关文档的完整集合。其中，程序是按事先设计的功能和性能要求执行的指令序列；数据是使程序能够正确地处理信息的数据结构；文档是与程序开发、维护和使用有关的图文材料。

在结构化程序设计时代，程序的最小单位是函数及子程序，程序与数据是分离的；在面向对象程序设计时代，程序的最小单位是类和接口，在类中封装了相关的数据及指令代码。

2. 软件的特性

当今已有的人工制品数不胜数，然而计算机软件却与任何传统的制造业产品不同，它具有许多突出的特性，概括如下。

(1) 形态特性

软件是无形的、不可见的逻辑实体，度量常规产品的几何尺寸、物理性质和化学成分对它是毫无意义的。

(2) 智能特性

软件是复杂的智力产品，其开发凝聚了人们大量的脑力劳动，产品本身也体现了人类的知识、实践经验和智慧，具有一定的智能。它可以帮助我们解决复杂的计算、分析、判断和决策问题。

(3) 开发特性

尽管已经有了一些工具（也是软件）来辅助软件开发工作，但到目前为止尚未实现自动化。软件开发中仍然包含了相当分量的个体劳动，使得这一大规模知识型工作充满了个性化行为和特征。

传统制造业的工艺都已经相当成熟，早已摆脱了手工作坊式的生产而大规模采用自动化的生产。虽然软件工作者希望软件的生产也能够像硬件生产那样基于已有的零部件进行组装，但很多软件产品是根据用户的需求进行定制开发的个性化产品。

(4) 质量特性

软件产品的质量控制存在着一些难于克服的实际困难，表现在以下方面：

- 软件的需求在软件开发之初常常是不确切的，也不容易确切地给出，并且需求还会在开发过程中出现变更，这就使软件质量控制失去了重要的可参照物。
- 软件测试技术存在不可克服的局限性。任何测试都只能在极大数量的应用实例数据中选取极为有限的数据，致使我们无法检验大多数实例，也使我们无法得到完全没有缺陷的软件产品。
- 已经长期使用或多次反复使用的软件没有发现问题，但这并不意味着今后的使用也不会出现问题。

这一特性提醒我们：一定要警惕软件的质量风险，特别是在某些重要的应用场合，需要提前准备好应对策略。

(5) 生产特性

与硬件或传统的制造业产品的生产完全不同，软件一旦设计开发出来，如果需要提供给多个用户，它的复制十分简单，其成本也极为有限。因此，软件产品的成本主要是设计开发成本，同时也不能采用管理制造业生产的办法来解决软件开发的管理问题。

(6) 管理特性

由于上述的几个特点，使得软件的开发管理显得更为重要，也更为独特。这种管理可归结为对大规模知识型工作者的智力劳动管理，其中包括必要的培训、指导、激励、制度化规程的推行、过程的量化分析与监督，以及沟通、协调，甚至是过程文化的建立和实施。

(7) 环境特性

软件的开发和运行都离不开相关的计算机系统环境，包括支持其开发和运行的硬件和软件。软件对于计算机系统的环境有着不可摆脱的依赖性。

(8) 维护特性

软件投入使用以后需要进行维护，但这种维护与传统产业产品的维护在概念上有着很大差别。如建筑物、机械和电子产品的维修大都是由于使用（也包括非正常使用）中造成了材料的老化、腐蚀或机械性磨损等，有待于通过维修恢复其功能或性能，而软件产品使用中出现的问题并非是使用造成的，也不是使用时间久形成的，软件维护往往是为了修正开发时遗留下来的、隐蔽的、那些在特定运行条件才暴露的缺陷，也可能是为了扩展与提升软件的功能或性能

以及为了适应运行环境的改变。

对软件的维护往往不能像硬件那样通过更换零件来解决，而需要对不适应的部分软件进行修改。

(9) 废弃特性

当软件的运行环境变化过大，或是用户提出了更大、更多的需求变更时，如果再对其实施适应性维护已不划算，那么软件将走到其生存期终点而被废弃（或称退役），此时用户将考虑采用新的软件代替。因此，与硬件不同，软件并不是由于“用坏”而被废弃的。

(10) 应用特性

软件的应用极为广泛，如今它已渗入国民经济和国防的各个领域，现已成为信息产业、先进制造业和现代服务业的核心，占据了无可取代的地位。

1.1.2　软件的分类

不同类型的工程对象，对其进行开发和维护有着不同的要求和处理方法，所以还找不到一个统一的严格分类标准。按照软件的作用，一般可以将软件做如下分类。

(1) 系统软件

系统软件是能与计算机硬件紧密配合在一起，使计算机系统各个部件、相关的软件和数据协调、高效地工作的软件。例如，操作系统、数据库管理系统、设备驱动程序以及通信和网络处理程序等。系统软件在运行时需要频繁地与硬件交互，以提供有效的用户服务、资源共享，其间伴随着复杂的进程管理和复杂的数据结构处理。系统软件是计算机系统必不可少的一个组成部分。

(2) 应用软件

应用软件是在系统软件的支持下，在特定领域内开发，为特定目的服务的一类软件。例如，在国民经济领域使用最广泛的商业数据处理软件、工程与科学计算软件、ERP 软件；计算机辅助设计/制造（CAD/CAM）、系统仿真、实时控制、智能嵌入产品（如汽车油耗控制、仪表盘数字显示、刹车系统）、人工智能（如专家系统、模式识别）等方面所使用的软件；事务管理、办公自动化、电子商务等方面的软件。

2000 年以后，由于互联网技术的发展，传统 C/S（Client/Server，客户机/服务器）结构的软件逐渐向 B/S（Browser/Server，浏览器/服务器）结构迁移，称为 Web 应用系统。同时，互联网软件（如门户网站、移动应用）大量出现，已经成为与我们的日常生活息息相关的应用软件，对传统商业模式造成了很大的冲击。

(3) 支撑软件

支撑软件亦称为工具软件，是协助用户开发软件的工具性软件，其中包括帮助程序人员开发软件产品的工具，也包括帮助管理人员控制开发进程的工具。支撑软件可分为纵向支撑软件和横向支撑软件。纵向支撑软件是指支持软件生存期某阶段特定软件工程活动所使用的软件工具，如需求分析工具、设计工具、编码工具、测试工具、维护工具等，横向支撑软件是指支持整个软件生存期各个活动所使用的软件工具，如项目管理工具、配置管理工具等。20 世纪 90 年代中后期发展起来的软件开发环境以及后来开发的中间件则可看成现代支撑软件的代表。软件开发环境主要包括环境数据库、各种接口软件和工具组，三者形成整体，协同支撑软件的开发与维护。

(4) 可复用软件

最初实现的典型的可复用软件是各种标准函数库，通常是由计算机厂商提供的系统软件的一部分。后来可复用的范围扩展到算法之外，数据结构也可以复用。到了 20 世纪 90 年代，作

为复用的基础，可复用的范围从代码复用发展到体系结构复用、开发过程复用。面向对象开发方法的核心思想就基于复用，为此，建立了可复用的类库、应用程序库，以及可复用的设计模式等。

其中的可复用成分称为可复用构件。在开发新的软件时，可以对已有的可复用构件稍加修改或不加修改，复用所需的属性或服务。

1.2　软件危机与软件工程

1.2.1　软件危机

20 世纪 60 年代，计算机已经应用在很多行业，解决问题的规模及难度逐渐增加，由于软件本身的特点及软件开发方法等多方面问题，软件的发展速度远远滞后于硬件的发展速度，不能满足社会日益增长的软件需求。软件开发周期长、成本高、质量差、维护困难，导致 20 世纪 60 年代末软件危机的爆发。

这些矛盾表现在具体的软件开发项目上，最突出的实例就是美国 IBM 公司在 1963 年至 1966 年开发的 IBM 360 机的操作系统。这个项目的负责人 F. D. Brooks 事后在总结开发过程中的沉痛教训时说："正像一只逃亡的野兽落到泥潭中做垂死的挣扎，越是挣扎，陷得越深。最后无法逃脱灭顶的灾难……程序设计工作正像这样一个泥潭……一批批程序员被迫在泥潭中拼命挣扎……谁也没有料到问题竟会陷入这样的困境……"

除了软件本身的特点，软件危机发生的原因主要有以下几个方面：

1）缺乏软件开发的经验和有关软件开发数据的积累，使得开发工作的计划很难制定。主观盲目地制定计划，往往与实际情况相差太远，致使常常突破经费预算，工期一拖再拖。而且对于软件开发工作，给已经拖延了的项目临时增加人力只会使项目更加拖延。

2）软件人员与用户的交流存在障碍，除了知识背景的差异，缺少合适的交流方法及需求描述工具也是一个重要原因，这使得获取的需求不充分或存在错误，存在的问题在开发初期难以发现，往往在开发后期才暴露出来，使得开发周期延长，成本增高。

3）软件开发过程不规范，缺少方法论和规范的指导，开发人员各自为战，缺少整体的规划和配合，不重视文字资料工作，软件难以维护。

4）随着软件规模的增大，其复杂性往往会呈指数级升高。

5）缺少有效的软件评测手段，提交用户的软件质量差，在运行中暴露出大量的问题，轻者影响系统的正常使用，重者发生事故，甚至造成生命财产的重大损失。

与 50 年前相比，当前的软件开发技术已经有了长足的进步，但由于目前软件要解决的问题越来越复杂和庞大，同时，用户对软件的质量和开发周期提出了更高的要求，因此软件开发人员依然面临开发周期长、成本居高不下、无法达到质量要求等问题。

1.2.2　软件工程

为了克服软件危机，1968 年 10 月在北大西洋公约组织（NATO）召开的计算机科学会议上，Fritz Bauer 首次提出"软件工程"的概念，试图将工程化方法应用于软件开发。

许多计算机和软件科学家尝试，把其他工程领域中行之有效的工程学知识运用到软件开发工作中来。经过不断实践和总结，最后得出一个结论：按工程化的原则和方法组织软件开发工作是有效的，是摆脱软件危机的一条主要出路。