HZ BOOKS
华章教育

高等院校计算机课程案例教程系列

本书为教师配有电子教案

软件工程
方法与实践

窦万峰 等编著

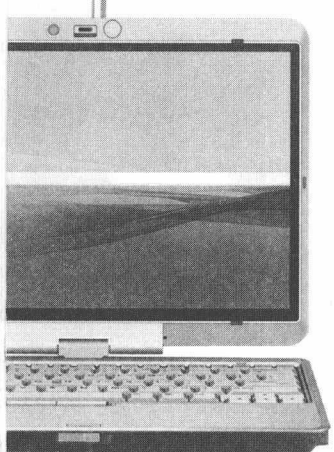
- 分别从结构化开发范型和面向对象开发范型角度循序渐进介绍软件开发过程相关的原理、方法和技术。
- 理论知识和案例分析相结合，以5个典型案例项目贯穿全书。

机械工业出版社
China Machine Press

高等院校计算机课程案例教程系列

软件工程
方法与实践

窦万峰 等编著



- 分别从结构化开发范型和面向对象开发范型角度循序渐进介绍软件开发过程相关的原理、方法和技术。
- 理论知识和案例分析相结合，以5个典型案例项目贯穿全书。

机械工业出版社
China Machine Press

软件工程学将计算机科学理论与现代工程方法论相结合，着重研究软件过程模型、分析与设计方法、软件工程开发与管理技术和工具，是指导软件生产和管理的一门新兴的、综合性的应用科学。本书分别从传统的结构化开发范型和面向对象开发范型两个方面，把软件工程的概念和理论知识融入到实践当中，通过丰富的案例分析与设计，更深入地理解软件开发中各个阶段的技术、方法和管理过程。本书包括软件工程与过程、软件需求分析与建模、软件设计、软件测试与维护和软件工程管理五个部分，共16章，深入介绍了软件开发“工程化”思想。

本书适合作为高等院校软件工程课程的教材，即适用于计算机专业的学生，也适用于其他非计算机专业从事软件开发与应用及管理的专业学生和技术人员学习的教材。本书还可以成为从事软件开发人员必备的参考书。

图书在版编目（CIP）数据

软件工程方法与实践 / 窦万峰等编著. —北京：机械工业出版社，2009.5
（高等院校计算机课程案例教程系列）

ISBN 978-7-111-26758-4

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（2009）第050788号

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：姚　蕾
北京慧美印刷有限公司印刷
2009年5月第1版第1次印刷
184mm × 260mm · 18.75印张
标准书号：ISBN 978-7-111-26758-4
定价：32.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换
本社购书热线：（010）68326294

前 言

软件工程概念自1968年提出以来，经过了近四十年的发展，为软件行业从业人员从事软件开发和维护提供了理论指导和基本原则，促进了软件产品和软件行业的快速发展，也促进了软件工程自身的理论体系的完善和发展。

软件工程涉及软件分析、设计、实现和维护等软件生命全过程，包含了一系列原理、方法和实践，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工程实践来开发和管理项目，并进行过程改进，促进软件企业向标准化和成熟化发展。

软件工程领域包括三个重要方面：软件开发、软件项目管理、过程改进。软件工程是一个理论与实践相结合的学科，更注重通过实践来理解理论和原理与方法。为此，本书结合作者多年从事软件工程教学和项目开发的经验，通过5个项目实例，从不同的角度和范型循序渐进介绍软件开发过程中所涉及的原理、方法与技术。

全书分为五个部分：

第一部分：软件工程概述。共4章内容，初步介绍软件工程的基本概念，涉及的主要内容，软件过程生命周期及其模型，以及本教材的案例与要求。

第二部分：软件分析。在这一部分共安排了3章内容，主要介绍软件分析与建模的原理和方法，包括结构化分析和面向对象分析的原理和方法。本部分分别介绍了分析的基本过程、非形式化的分析方法和获取需求的策略；介绍了结构化分析建模技术，包括面向数据流的分析技术和面向数据的分析技术；介绍了面向对象分析建模技术，包括用例建模、对象建模等，介绍如何用UML语言表示面向对象模型。本部分用案例进一步深化分析的思想和原理及过程。

第三部分：软件设计。在这一部分共安排4章内容，主要介绍软件设计原理和方法，包括结构化设计和面向对象设计技术。介绍设计的基本概念、原理和过程；介绍面向数据流的设计方法和面向数据结构的设计方法；介绍面向对象的设计方法，包括设计模型、精化设计、动态设计；介绍详细设计和界面设计等。本部分通过案例深入理解设计方法和过程。

第四部分：软件测试与维护。在这一部分安排两章内容，主要介绍软件测试原理与技术、维护策略与方法。介绍测试过程、测试基本技术、测试用例设计；介绍软件维护类型、可维护性和结构化维护过程。

第五部分：管理。在这一部分共安排3章内容，主要介绍软件项目管理概念与原理、软件成本估算、项目计划与管理。介绍软件项目管理的概念和指导原则、项目管理过程和任务、项目管理的范围等；介绍软件项目成本估算方法，包括项目度量、分解技术、经验模型；介绍项目计划安排、进度安排与监控、软件配置管理、风险管理等。

本书内容翔实，提供5个典型案例支持，便于读者学习和深入体会软件工程的原理和思想。不同的案例充分体现了不同的技术，因此有区别的案例介绍和分析层次会更加突出方法的实用性。本书第1～3章由宋效东编写，第4～9和11章由窦万峰编写，第10章由彭涛编写，第12章由洪奎编写，第13～14章由李芷编写，第15～16章由吴怀岗编写。

本书适合作为高等院校软件工程课程的教材，既适用于计算机专业的学生，也适用于其他非计算机专业从事软件开发与应用及管理的专业的学生。本书还可以作为软件开发人员的参考用书。

由于作者水平有限，难免有疏漏之处，恳切各位读者指正。尤其是针对提供的案例的详细程度和方案的多样性，请读者给予意见，以便以后改进和完善。

窦万峰

2009年2月

教学建议

本书分为五个部分，其中第一部分总体介绍软件工程的发展、涉及的内容和主要的技术与方法。通过第一部分的学习，可以整体了解软件工程课程的概貌，比如软件工程涉及技术和管理两个方面（第1章），软件工程的工程化思想和基本原理（第1章），软件工程有传统结构化范型和面向对象范型两大体系（第1章），掌握软件开发的工程化开发过程（第2章），并深入学习软件工程过程模型（第3章），最后还引入了本书将要深入介绍的开发案例（第4章），并站在用户的角度给出案例系统的用户需求。第一部分的重点内容是软件工程的工程化思想和基本原理、软件工程过程模型。教师根据课时可重点讲述第1章和第3章的内容，而这些内容的思想将会在后续章节中逐步体现和应用。

第二、三、四部分按照软件开发基本过程进行设置，即软件开发的分析、设计、实现、测试和维护五个基本过程，这几个部分主要从技术与方法的角度介绍了软件开发所涉及的方法，并分别从传统的结构化和面向对象两大范型进行介绍。教师可根据实际情况选择某个范型进行讲述，本书在这两个范型中有所侧重地使用不同案例介绍软件工程的基本方法和实践。

第二部分是软件需求分析的内容。第5章介绍了软件需求分析的基本概念、原理与过程，教师可根据实际情况简略讲述，第6章和第7章分别从传统结构化范型和面向对象范型不同角度重点介绍结构化分析技术与方法和面向对象分析技术与方法，并用翔实的实例和案例深入介绍这些技术与方法的应用过程。

第三部分涉及软件设计的内容。第8章介绍软件设计的基本概念、原理和思想，这些内容是学习的重点。当然这些内容的掌握需要第9章和第10章的进一步展开介绍，并使用实例来理解软件设计的技术与方法。第11章从详细设计和实现的角度讲述软件模块或组件的内部设计。教师可根据实际情况选择某个范型进行讲述，并根据时间对内容有所选取。

第四部分是有关软件测试与维护的内容。第12章的前半部分主要介绍了软件测试的原理与过程，后半部分通过一些实例介绍测试的基本方法，后半部分是学习的重点。第13章介绍了软件交付以后的维护问题，主要从原理的角度作了介绍，可以作为了解内容学习。

第五部分介绍软件工程管理。该部分分别介绍了软件工程管理的基本概念与过程、软件项目估算和软件项目计划三个方面。第14章从基本概念的角度介绍软件工程管理的基本概念、管理过程和管理内容。第15章从技术的角度介绍了软件项目估算的方法，通过实例介绍估算的技术与方法及步骤，是学习的重点。第16章介绍了质量度量和项目计划安排与调度，重点需要掌握项目计划的基本方法。

书中在介绍技术与方法的章节都重点给出相关案例的分析，帮助学生深入理解方法的实施关键点。每一章后面都有相关的习题，通过练习来检验和加深对本部分内容的掌握情况。

窦万峰

2009年2月

目 录

第三部分 软件设计与建模

第四部分 软件测试与维护

第五部分 软件工程管理

第一部分

软件工程与过程

本部分将介绍软件工程的基本概念和软件过程及其模型，将回答以下问题：

- 什么是软件工程？
- 什么是工程化思想？
- 什么是软件过程？有哪些过程模型？
- 如何建立过程模型？
- 什么是统一过程？
- 什么是敏捷过程？有哪些模型？
- 什么是软件工程实践？

第1章 概　述

软件工程（Software Engineering，SE）的概念是在20世纪60年代末期提出的。这一概念的提出，目的是倡导以工程的原理、原则和方法进行软件开发，用来解决当时出现的“软件危机”。

B. W. Boehm为软件工程下的定义为：“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为软件工程下的定义为：“软件工程是为了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。”

1983年IEEE（国际电气与电子工程师协会）提出了IEEE软件工程标准术语，将软件工程定义为：“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、相关的文件资料以及在计算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，且又有一些学者提出了更完善的定义，但其主要思想都是在强调在软件开发的过程中应用工程化思想的重要性。

软件工程的目标是：根据需求分析确定可行性后，在给定的时间内开发出具有可修改性、有效性、可靠性、可维护性、可重用性、可适应性、可移植性、开销合宜并满足用户需要的软件产品。

1.1 软件工程的基本概念

软件工程的主旨是以工程化的思想进行软件开发，以生产高质量和高效率的软件，也就是说，软件工程研究的基础就是软件。那么，软件是怎么定义的？又有哪些特性？

1.1.1 软件的角色和特性

既然软件工程中的主角是软件的开发，那么在现代社会中，软件究竟担任一种什么样的角色？我们使用的大部分软件同时担任着两个角色，一个是产品，同时又是工具。利用软件，我们可以存储信息，进行信息的变换。当然，这里的信息并不仅仅是数字，也可以是多媒体资料。

软件（software）是计算机系统中与硬件相对应的另一部分，包括一系列程序、数据及其相关文档的集合。在这里，**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计算机程序的数据结构；**文档**是与程序开发、维护和使用有关的图文资料。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。

要理解软件的真正含义，首先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出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却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：

1）复杂性：软件是一个庞大的逻辑系统，比任何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，软件主要依靠人脑的“智力”构造出来，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。

2）一致性：软件必须和运行软件的硬件保持一致，这是由软件对硬件的依赖所决定的，虽然可以用硬件顺应软件，或者用软件顺应硬件来保持一致，但一般都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，则软件必须和现有硬件系统接口。还有一种情况，因为计算机的软件和硬件是具有功能互换性的，所以也可能用软件来替代硬件接口的功能。

3）软件不存在磨损和老化问题：一般的器械设备在运行的周期中，其失效率大致遵循图1-1所示的U型曲线（浴盆曲线）。而软件的情况与此截然不同，因为它不存在磨损和老化的问题。事实上，

软件不会磨损，但它却会退化，因此，软件在其生命周期中，一般都需要进行多次维修。图1-2中的理想曲线是软件的实际故障曲线粗略的简化。

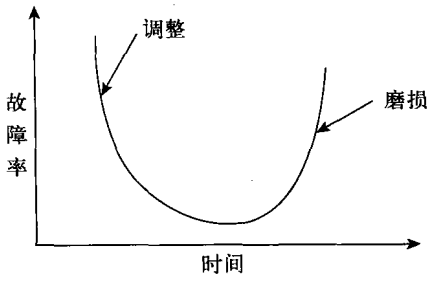


图1-1 硬件的故障曲线

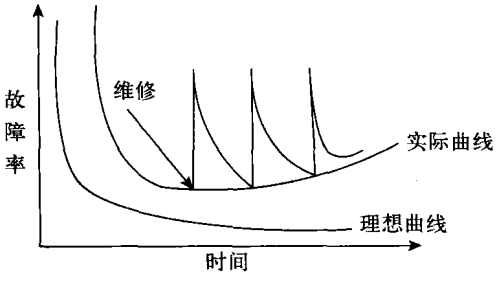


图1-2 软件的理想故障曲线和实际故障曲线

4）易变性：软件在生产过程中甚至在投入运行之后，也还可以再改变。软件必须经历变化并容易改变，这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功效；修改软件毕竟比更换硬件容易，使得软件易维护、易移植、易复用。因为修改软件的压力总是存在的，所以软件产品始终在“变”。这种动态的变化不仅难以预测、难以控制，而且可能对软件的质量产生负面影响。

5）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，这里就牵扯到软件的可移植性。好的软件在设计时就考虑到软件如何应用到不同的系统平台。

6）软件的开发是一个复杂的过程，而且软件的开发成本非常高。

1.1.2 软件演化

按照系统论的观点，演化是事物从一种多样性统一形式转变成另一种多样性统一形式的具体过程。软件的发展也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。我们知道软件就是计算机程序及说明程序的各种文档。纵观前后几十年间，软件的演化大致经历了四个阶段：

第一代：从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生产方式。

第二代：从20世纪60年代初到20世纪70年代初，是计算机软件发展的第二个阶段，我们称这个时期为程序系统阶段。在这个阶段，诞生了软件工程学科。程序的规模已经发展得很大，需要多人分工协作，软件的开发方式由个体生产发展到了小组生产。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。因此，许多软件产品后来根本不能维护，最终导致出现了严重的软件危机。

第三代：从20世纪70年代中期至20世纪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到软件的开发过程中，用工程化的原则、方法和标准来开发和维护软件。

第四代：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，促进了软件产业的飞速发展，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。

早在软件刚诞生时，人们普遍认为编程是高不可攀的技术领域。当然，当时的软件的性能也不可与当今的软件相提并论。随着计算机的普及，程序的稳健性和易读性受到了广泛的关注，于是，程序从个人按自己意图创造的“艺术品”转变成了能被广大用户接受的工程化产品。由于外部环境和用户需求的不断变化及软件开发技术的不断发展，注定了软件系统只有不断的演化才能适应用户的新需求。

从整个系统的角度看，开发软件系统的目的是满足用户的需求，提高生命力。因此，软件的需求仍是软件发展的动力。早期的程序开发者只是为了满足自己的需要，这种自给自足的生产方式是其低

级阶段的表现。进入软件工程阶段以后，软件的开发具有社会属性，它要在市场中流通以满足更多用户的需要。

软件演化过程的各个阶段也有不同的特征，在这些阶段中，软件工作的范围从只考虑程序的编写扩展到涉及整个软件生存周期。软件演化过程包括演化计划、软件理解、软件需求变更的分析、重构、测试等阶段。

1.1.3　软件神化和危机

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，甚至有些人曾误解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，但是，旧的观念和习惯不可能轻易地被丢弃，仍有部分人认同软件神化的思想。

在软件技术发展的第二阶段，随着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，导致问题积累起来，形成了日益尖锐的矛盾。这就导致了软件危机。

这场软件危机主要表现在：软件的规模越来越大，复杂度不断增加，软件的需求量也日益增大，且价格昂贵，供需差日益增大。而软件的开发过程是一种高密集度的脑力劳动，软件开发常常受挫，质量差，很难按照指定的进度表来完成预定的任务，软件的研制过程很难管理，即软件的研制往往会失去控制。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入市场，部分软件花费了大量的人力财力，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，历时五年，但最终还是失败，导致伦敦股票市场声誉下跌。我们称软件开发和维护过程中所遇到的这种严重问题为软件危机。

软件危机主要是两个方面的问题：如何开发软件，以满足客户对软件日益增长的需求；如何维护数量不断膨胀的现有软件。

一般情况下，在开发一个新型计算机系统或修改一个现有系统的过程中，资金主要是花费在软件系统的开发方面。由于计算机系统发展的早期所形成的错误概念与做法，导致了计算机软件的开发严重受阻。最让软件工程师头痛的是，有些用错误方法开发的软件几乎根本无法维护，结果是提前宣布软件开发的结束。

1.1.4　软件危机的解决途径

在软件危机相当严重的背景下，产生了软件工程的思想。在引入工程化的思想后，人们分析了导致软件危机的原因，并提出了解决的对策。

1）在软件开发的初期阶段，需求提得不够明确，或是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，造成开发后期矛盾的集中暴露。因此，前期工作很重要。前期的需求分析不到位的话，认为软件的开发仅仅是编写程序，就很可能导致后期开发的软件达不到客户的要求，不得不进行软件的二次开发。并且，需求分析可以进一步地了解客户的需求，对软件的开发很有帮助。

2）需求分析后，要做好软件定义时期的工作，这样可以在一定程度上降低软件开发的成本，同时又在无形中提高了软件的质量，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。

3）开发过程要有统一的、公认的方法论和规范指导，参加的人员必需按照规定的方法论进行开发。重视设计和实现过程的资料，不要忽视每个人的工作与其他人的接口，否则开发出的软件将很难进行维护。由于软件是逻辑部件，开发阶段的质量较难衡量，开发质量较难评价，开发过程管理和控制较难，这就需要开发人员一定要有统一的软件工程理论来指导。

4）必须在测试阶段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，重视有关软件开发数据的积累，确保开发工作的计划按时完成，在期限内完成软件的开发。

1.2 工程化思想

软件工程化思想的核心是，把软件看作是一个工程产品，这种产品的完成需要经过需求分析、设计、实现、测试、管理和维护几个阶段。要用完善的工程化原理研究软件生产的规范方法，这样，软件的开发不仅会在指定的期限内完成，还会节约成本，保证软件的质量。

1.2.1 工程化所涉及的范围

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去进行软件开发、维护和管理的学科。因此，软件工程学涉及的范围也很广，包括计算机科学、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技术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。

1.2.2 工程化管理思想

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品，它创造的唯一产品或服务是逻辑载体。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。软件产品的生产不同于其他产品的制造，软件过程更多的是设计过程（没有制造过程）。另外，软件开发不需要使用大量的物质资源，而主要依靠人力资源。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，软件项目管理与其他项目管理相比，有很大的独特性。

软件开发中工程化的思想主要体现在软件项目管理方面。软件项目管理的作用一方面是提高质量，降低成本；而另一方面则是为软件的工程化开发提供保障。与其他项目相比，软件项目还是一种比较新兴的领域，在软件行业的迅猛发展中，一些问题和危机逐渐暴露出来，如：项目时间总是推迟、项目结果不能令客户满意、项目预算成倍超支、项目人员不断流动等都是软件开发商可能面临的问题。

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，管理者主要是在应付突发事件，对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。

为了解决软件工程中的多种问题，美国软件工程研究所（SEI）自1986年开始研究软件过程成熟框架，并于1991年提交了能力成熟度模型（Capability Maturity Model）CMM V1.0版。之后历经多方软件专家的评审，SEI又发布了CMM V1.1版，并更名为SW-CMM。1999年底发布了SW-CMM V2.0版。该模型强调企业软件开发能力取决于企业的过程能力而不是个人能力，强调持续的过程能力的改善是衡量软件企业软件开发管理水平的重要参考。该模型既可以作为软件开发组织改善软件开发过程的参考模型，也可以作为用户评估软件项目承包商的依据。

如今，软件开发的工程化管理思想已经得到了认可，软件的开发管理已经不同于以往过分依赖软件技术精英。运用项目管理的经验和方法是软件项目成功的前提和保证，这已是今天的软件业内人士的共识。随着信息技术的飞速发展，软件产品的规模也越来越庞大，个人单打独斗的作坊式开发方式已经越来越不适应发展的需要。尤其是近几年，随着网络技术的快速发展，项目管理也随之快速发展，各软件企业都在积极将软件项目管理引入开发活动中，对开发实行有效的管理，并都取得了不同的成绩。

1.3 软件工程两大范型

范型的意思是指模型或模式，而在软件工程学科中，范型用来表示一套涵盖整个软件生产过程的技术的集合。目前使用得最广泛的软件工程方法学，分别是结构化范型和面向对象范型。

1. 传统的结构化范型

结构化范型自1968年被提出以来经过了多年的发展，形成了一套完整的体系。构成结构化范型

的技术包括结构化分析、结构化设计、结构化编程和结构化测试，这些技术在以数据为主或小型系统方面得到广泛应用。采用结构化技术来完成软件开发的各项任务，并使用适当的软件工具或软件工程环境来支持结构化技术的运用。这种范型把软件的生命周期依次划分为若干个阶段，然后顺序地完成每个阶段的任务。采用这种范型开发软件的时候，从对问题的抽象逻辑分析开始，一个阶段接一个阶段地进行开发，从而降低了整个软件开发工程的困难程度。引入结构化范型所带来的主要改进是在软件工业上。在使用结构化范型之前，大多数的软件组织都不使用任何特定的技术，每个人都以自己的方式工作，即所谓的个人英雄主义。结构化范型在工业领域的运用是软件工程被大规模采纳的主要原因。

结构化范型获得采纳和推广的原因是，结构化技术要么面向行为，要么面向数据，但没有既面向数据又面向行为的。软件的基本组成部分包括产品的行为和这些行为操作的数据。有些结构化技术（如数据流分析）是面向行为的，这些技术集中处理产品的行为，数据则是次要的。反之，有些技术（如JSP系统开发技术）是面向数据的，这些技术以数据为中心，在数据上操作的行为则是次要的。

随着软件产品规模的不断增大，结构化范型有时不能应付。也就是说，结构化技术处理5000行或50000行代码是有效的。然而，当今的软件产品，有50000行或5000000甚至更多行代码的产品非常普遍。维护阶段是结构化范型不能满足人们期望的第二个方面。结构化范型在当年前得以发展的主要动力是，软件维护的费用占到软件费用的2/3。然而，结构化范型未能很好地解决这一问题。

2. 面向对象范型

面向对象范型把数据和行为看成同等重要，即将对象视作一个融合了数据及在其上操作的行为的统一的软件组件。对象的概念符合业务或领域的客观实际，反映了实际存在的事物，也符合人们分析业务本质的习惯。

面向对象技术自20世纪90年代提出以来得到快速发展，并被运用到各种各样的软件应用开发中。面向对象技术将数据和数据上的操作封装在一起，对外封闭这些细节，从而实现了信息隐藏的目的。使用这个对象的用户只需要知道其暴露的方法，通过这些方法来完成各种各样的任务，完全不需要知道对象内部的细节，保证相对独立性。

面向对象的优势主要体现在维护阶段。相对于结构化技术，无论对象的内部细节如何变化，只要对象提供的方法即接口保持不变，则整个软件产品的其他部分就不会受到影响，不需要了解对象内部的变化。因此，面向对象范型使维护更快、更容易，同时产生回归的机会也大大降低了。

面向对象范型使开发变得相对容易。大多数情况下，一个对象对应物理世界一个事物。软件产品中的对象和现实世界的同等对应物之间的密切对应关系，促进了更优化的软件开发。对象是独立的实体，因此面向对象促进了复用，降低了开发维护的时间和费用。

目前，传统的结构化范型仍然是人们在开发软件时使用得十分广泛的软件工程方法学。广大软件工程师对这种范型比较熟悉，而且在开发某些类型的软件时也比较有效，因此，在相当长一段时期内这种方法学还会有生命力。此外，如果没有完全理解结构化范型，也就不能深入理解这种范型与面向对象范型的差别以及面向对象范型为何优于结构化范型。

在使用结构化范型时，分析阶段和设计阶段的过渡太快；而面向对象范型以迭代的方式从一个阶段向另一个阶段过渡，比结构化范型平滑得多，从而降低了开发过程中的故障数。

1.4　软件工程思想与基本原理

自从1968年提出“软件工程”这一术语以来，研究软件工程的专家学者们陆续提出了100多条关于软件工程的准则或信条。美国著名的软件工程专家B. W. Boehm综合这些专家的意见，并总结了多家公司开发软件的经验，于1983年提出了软件工程的七条基本原理。Boehm 认为，这七条原理是确保软件产品质量和开发效率的原理的最小集合。它们是相互独立的，是缺一不可的最小集合；同时，它们又是相当完备的。下面简要介绍软件工程的七条原理。