高等学校计算机专业“十一五”规划教材

软件工程
—— 理论、方法与实践

吴军华　主编

西安电子科技大学出版社
http://www.xduph.com

高等学校计算机专业“十一五”规划教材

软件工程——理论、方法与实践

吴军华　主编
赵艳红　段　江　等编著

西安电子科技大学出版社

内 容 简 介

本书主要从面向对象的角度阐述了软件工程的相关理论和方法。全书主要以 UML 为建模语言，以 UML 的发起人 Booch、Rumbaugh 和 Jacobson 建议的面向对象的分析和设计方法为核心内容，参照 IEEE 的软件工程知识体系，系统阐述了软件工程活动的理论、方法和技术。

本书第 1、2 章介绍了软件工程相关概念和过程活动；第 3 章讨论了基于 UML 的面向对象系统建模方法；第 4、5 章讨论了需求工程活动和面向对象的需求分析方法；第 6、7 章详细阐述了软件系统设计原则及软件体系结构设计，并重点讨论了面向对象的设计方法；第 8、9 章讨论了目前广泛用于软件系统设计的分布式体系结构和系统复用技术；第 10 章阐述了软件活动中的形式化模型定义方法；第 11、12 章讨论了软件编码以及测试活动和方法；第 13 章介绍了软件交付后的维护工作；第 14、15 章介绍了软件工程过程中的项目管理活动和过程改善技术；第 16 章简单介绍了净室软件工程技术。

本书可作为计算机及信息类专业本科生的教材，也可作为研究生和其他软件技术人员的学习参考书。

图书在版编目(CIP)数据

软件工程——理论、方法与实践/吴军华主编. —西安：西安电子科技大学出版社，2010.9

高等学校计算机专业“十一五”规划教材

ISBN 978-7-5606-2453-2

Ⅰ. ① 软… Ⅱ. ① 吴… Ⅲ. 软件工程—高等学校—教材 Ⅳ. ① TP311.5

中国版本图书馆 CIP 数据核字(2010)第 120343 号

策　　划　臧延新

责任编辑　杨宗周　臧延新

出版发行　西安电子科技大学出版社(西安市太白南路 2 号)

电　　话　(029)88242885　88201467　　邮　　编　710071

网　　址　www.xduph.com　　电子邮箱　xdupfxb001@163.com

经　　销　新华书店

印刷单位　西安文化彩印厂

版　　次　2010 年 9 月第 1 版　2010 年 9 月第 1 次印刷

开　　本　787 毫米×1092 毫米　1/16　印张 17.875

字　　数　420 千字

印　　数　1～3000 册

定　　价　26.00 元

ISBN 978 - 7 - 5606 - 2453 - 2/TP • 1223

XDUP 2745001-1

高等学校计算机专业“十一五”规划教材

编审专家委员会

前　言

无庸置疑，计算机软件已在国民经济和社会发展中占据了极为重要的地位。软件的广泛应用、Internet的快速发展、一些关键领域对软件的特殊要求，使得软件系统的规模和复杂性日益增加，软件开发方法和技术面临不断挑战。

开发大型软件需要面对软件的成本控制、可靠性需求、功能要求等诸多压力，犹如建造高楼大厦，需要有效的过程管理、科学规范的方法和有效的工具支持才能完成。运用工程化的方法开发软件是保证软件项目顺利实施的有效途径。软件工程旨在通过采用系统、规范、量化的方法指导软件开发，以在有效成本范围内产生高质量、满足用户需求的软件。

软件工程的概念因软件危机在1968年NATO会议上首次被提出。40年来软件工程发展迅速。SWECC(Software Engineering Coordination Committee)于本世纪初制定了软件工程知识体系(Software Engineering Knowledge Body)和软件工程教育体系(Software Engineering Education Body)，为软件工程活动和教育提供了指导性方向。实践性强、知识更新快、多学科交叉是软件工程学科所具有的特点。

通用的软件工程活动一般包括需求分析、设计和实现、软件验证和软件演化。一系列用于活动指导的分析、设计方法，模型描述机制，测试技术、开发规则等构成了软件工程方法。在相应方法和工具支持下的软件工程活动的集合构成了软件过程。

软件工程方法主要经历了两大阶段：结构化方法(Structural Method)阶段和面向对象方法(Object-Oriented Method)阶段。传统的结构化方法源于结构化程序设计语言的出现，曾经在软件工程领域很长时间内占据主导地位。20世纪80年代末随着面向对象语言的大量出现，软件工程方法和技术也经历了重大变革，Booch、OMT和OOSE等面向对象的分析和设计方法不断涌现，目前已逐步进入主要运用面向对象方法进行软件分析、设计和其他开发活动的阶段。各种面向对象的方法各有优势，方法间也存在着一定的差异，为开发过程的交流和统一带来困难。因此Booch方法、OMT和OOSE方法的发起人Booch、Rumbaugh和Jacobson综合了他们各自的模型描述方法，形成了统一建模语言UML(Unified Modeling Language)，目前已成为面向对象系统开发的通用建模语言。

本书主要以统一建模语言UML为主要建模工具，以面向对象的方法为核心阐述软件工程的主要知识点。和大多软件工程教材不同，本书注重软件工程领域新技术的发展，引入了软件体系结构、设计模式、分布式系统结构、组件技术、净室技术等代表软件先进技术的内容。书中分析和设计的示例为作者参与的项目，有较好的示范性。

全书共分为16章，建议授课学时40～52学时，第8、9、10、16章内容本科生可选学。

书中第1、4、6、8、9章由吴军华编写，第3、5、7、10、11章由赵艳红编写，第12～16章由段江编写，第2章由周晶编写，同时感谢刘继红、王栋、高春贞硕士为本书的绘图所做的工作。

由于作者水平有限，书中内容难免存在不足，欢迎广大读者批评和指正。

编　者

2010年6月

目　录

第1章 导 论

目前，各种计算机系统已广泛用于国民经济、国防建设、社会发展和人们的日常生活中。人们在各项活动中越来越多地依赖于计算机系统。在各类系统中，无论是大型的科学计算处理系统还是日常生活中常见的掌上电脑，计算机软件成本所占比例越来越大。同时计算技术及网络技术的飞速发展也使软件的本质和模型在不断变化，软件系统也越来越庞大和复杂，这使得在软件开发过程中，对成本、进度和软件质量的控制日渐困难。因此采用高效的软件开发技术，提高软件系统开发的效率和质量在软件开发过程中越来越重要。软件工程(Software Engineering)在1968年NATO(北大西洋公约组织)会议上因软件危机首次提出，目前已成为一门指导软件系统开发和维护的新兴工程学科，它运用计算机科学、工程科学、数学、管理学和社会学等原理和方法指导软件系统的开发过程，以实现高质量、有效的软件系统。

1.1 软 件

尽管在不同的软件发展阶段，人们对软件的认识有所不同，但目前对软件的定义已达成共识。软件的定义可以表述为：当运行时，能够提供所要求的功能和性能的计算机程序(Program)的集合；用于设置程序的配置数据(Configuration Data)和用于描述系统结构和功能以及使用系统的文档。软件由两大部分组成：一是机器可执行的程序及有关数据；二是机器不可执行的、与软件开发、运行、维护和使用有关的文档。

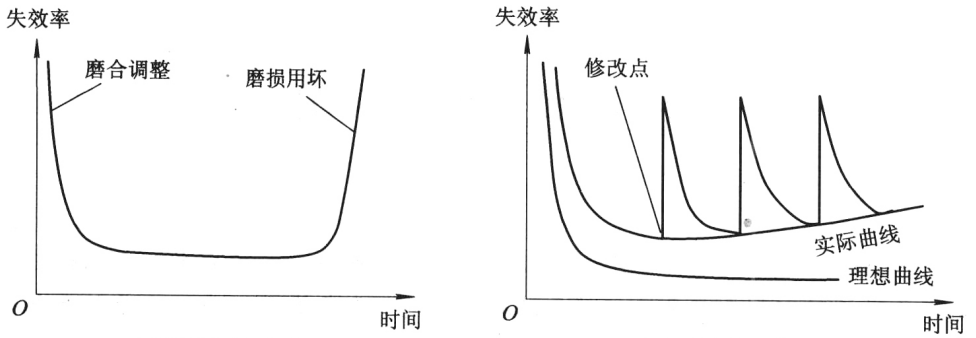
软件不是物理产品而是逻辑产品，在开发、维护方面与硬件产品相比有着完全不同的特性：

(1) 软件是设计开发的，而非传统意义上生产制造的。软件和硬件均可通过设计获得好的产品品质，但硬件制造阶段的质量问题是易于控制和纠正的，而软件生产阶段的质量控制要困难得多。相比硬件的生产，软件开发过度依赖于开发人员的素质、能力及协作，软件项目管理过程不能像硬件制造那样进行。

(2) 软件不会磨损。随着时间的推移，灰尘、振动、不当使用、温度等因素的影响，硬件的失效率会增大，这称之为磨损。而软件则不会受上述环境的影响。硬件和软件的失效率曲线如图1.1所示。

理论上讲，软件在交付初期失效率较高，随着缺陷的纠正，曲线将趋于平缓，也即软件不会磨损。但实际上，软件存在着退化，因为软件在交付运行后，会面临变更(所谓变更，是指软件因本身的缺陷需要纠正，新的需求需要满足，新的环境需要适应，而使软件要做修改和更新)。每次变更可能引入新的错误，使失效率在变更后陡然上升，一次又一次

的变更将会使软件退化。



(a) 硬件失效率曲线　　(b) 软件失效率曲线

图 1.1　硬件和软件失效率曲线对比

磨损的硬件可以使用备用件替换，退化的软件则没有替代品，因此软件交付后的维护过程是一个复杂的过程。

(3) 大多数软件是根据客户的要求定制的。硬件通常是依据一定规范标准来制造的，软件则多数是根据客户的要求定制的。虽然目前商业化软件的组件技术发展很快，但完全使用现成的组件实现软件系统仍不现实，基于组件的软件开发模式仍然需要根据软件需求来开发系统。

1.1.1　软件的发展

1942 年，伴随着第一台计算机的诞生，计算机程序开始出现。通常，软件发展至今可被划分为四个阶段。

第一阶段，20 世纪 50 年代初至 60 年代初是计算机系统发展的初期，因计算机设备庞大和价格昂贵，故主要应用于科学计算。计算机软件多为使用者自己编程，程序规模较小。由于硬件的限制，编程者往往注重程序的效率而不是程序的可理解性。软件开发过程不够规范，很少采用工程开发的思想，生产率较低，已出现成本、进度难以控制的局面。大多数软件系统采用批处理方式运行，已出现了 Algol、FORTRAN 等较早的高级编程语言和子程序库。

第二阶段，20 世纪 60 年代中期至 70 年代中期，此时计算机的应用已扩展至许多非数值计算领域，操作系统、结构化语言 Pascal、Cobol 和关系数据库系统、多用户、人机交互等概念出现。软件规模可达数万行代码，软件的概念已不仅是程序，出现了软件车间和专门的软件开发人员开发软件产品。但由于缺乏有效的软件工程方法和技术，软件内部逻辑的复杂性使得软件的理解和维护工作十分困难，20 世纪 60 年代末软件危机十分严重，导致 1968 年软件工程概念的首次提出。

第三阶段，20 世纪 70 年代中期至 80 年代，微处理器的出现，计算机网络和分布式技术走进应用，使软件更广泛地应用于各行各业，个人计算机更是走进了人们工作、生活的各个领域。各类软件公司得到发展，软件产品销售额不断增加。软件系统的规模、复杂性的增加和在重要领域的一些应用促进了软件开发过程的工程化，软件工程受到广泛重视，软件品质有了很大的提高。标志着新的软件开发思想的面向对象语言开始出现，主要以

Smalltalk、C++为代表。

第四阶段，20世纪80年代至今，面向对象技术已日益成熟，成为主流的软件开发方法。Internet技术迅速发展，今日已无处不在，基于Web的应用已成为一种极为重要和普遍的应用软件形式。异构和分布式系统的集成技术已成为主要的软件开发模式，因此出现了各类组件和分布式系统集成技术，如Sun公司的EJB/J2EE、Microsoft的COM/DCOM以及OMG的CORBA等。

1.1.2 软件的类型

软件从功能角度分为系统软件和应用软件，从服务对象的角度分为通用软件和定制软件。

系统软件(System Software)是服务于其他程序的程序集，软件一般与计算机软、硬件资源紧密结合，使计算机系统各软、硬件能协调高效地工作。系统软件通常具有频繁与硬件交互、为用户提供大量服务、进行有效的资源管理和进程管理、处理复杂数据结构等特点。软件通常隐藏了计算机系统底层资源的特征或实现细节，使用户可以方便、有效地使用这些资源。操作系统、数据库管理系统、计算机通信和网络软件、各种语言编译和解释程序、文件编辑器、系统诊断软件等均属于系统软件。

应用软件(Application Software)是在系统软件的基础上，为解决特定领域应用问题，为特定目的服务的一类软件。应用软件涉及广泛，科学计算软件、工业自动化控制系统、嵌入式系统软件、事务处理软件、人工智能和专家系统、系统仿真、办公自动化等均属于应用软件。

不同应用领域的软件也有其不同的特点，科学计算软件以数值算法为基础，通常计算的数据量较大，以往多采用FORTRAN语言进行程序设计，现在C语言、Ada语言已成为重要的科学计算语言。由于这类应用软件有着长期的积累，已形成很多的程序库和软件包。工业自动化软件通常与工业控制紧密相关，通常具有实时监控的特点，要求对监控的事件响应及时，且具有高可靠性和安全性，一般采用汇编语言、C语言和Ada语言来实现。嵌入式软件是嵌入进某个应用系统的计算机系统的软件部分，大型的嵌入式软件如航空航天系统、军事指挥控制系统，小型的如家用电器控制、汽车制动等。嵌入式软件一般和仪器、仪表、传感器等相连，要求具有实时特性的嵌入式软件同自动控制软件相似，目前嵌入式软件的开发主要基于一些嵌入式平台。事务处理和办公自动化软件主要涉及大量的事务数据处理，也被统称为管理信息系统(MIS)。管理信息系统通常以数据库管理系统为平台，采用四代语言来实现，大多具有良好的人机交互界面。人工智能和专家系统，其目标在于模仿人的智能，这类软件在求解复杂问题时，不是采用传统的计算或分析方法，而是采用规则、演绎推理等方法以及知识库等技术。LISP和Prolog是这类系统最多被使用的程序设计语言，目前人工智能、机器学习是计算机领域研究和发展的热点。

Web应用软件是Internet环境下的应用系统，随着电子商务和B2B应用的不断发展，Web应用程序的计算环境和规模在不断增加，通常与企业数据库和业务规则有着紧密的联系。

依据软件的通用性和特定性，又可将软件分为通用软件和定制软件。这两种类型的软件在开发过程中也会有一定的区别。

通用软件(Generic Products)是由开发机构生产的相对独立的在市场出售的软件产品，可以满足一般用户的一些共性需求。如数据库软件、文字处理软件、图形包、建模工具等。通用软件一般受众面巨大，软件的使用周期也较长，需要良好的可维护性和可靠性，因此软件开发过程中，应有良好的测试和确认过程。

定制软件(Bespoke Products)是为特定用户开发的软件产品，通常根据与用户的合约开发。如特定的电子商务系统、交通控制系统等。定制软件更关注特定用户的满意度。

1.1.3 软件质量特性

质量特性是“产品、过程或体系与需求有关的固有特性”。软件质量特性反映了软件的本质。

软件是一个逻辑部件，而不是一个物理部件，具有抽象性。因而软件与其他工程对象有着明显的区别，即软件是无形的，必须通过观察、分析、思考或运行才能获知其功能、性能及其他特性。这给软件的设计、生产和管理带来许多困难。软件是复杂的，软件的复杂性不仅来自于它所要解决的实际问题的复杂性，还涉及程序逻辑结构的复杂性和对软件人员较高的技术要求。因此长期以来，软件复杂性的增长与软件技术的发展存在差距。

软件的开发过程更多体现的是人的智力发挥过程。软件通常需要几个人共同协作，而人与人之间的通信是要付出代价的，除了时间上的开销，同时通信中的信息不准确常常会使错误增加。因此软件质量的控制和硬件质量之间有许多共同点，也存在很大的差异。

软件质量的定义有多种，美国国家标准学会(ANSI)在其 ANSI/ASQCA3/1978 标准中将软件质量定义为：“软件质量是软件产品或服务的特性和特征的整体，它取决于软件满足给定需求的能力”。具体来说，软件质量就是软件符合明确描述的功能和性能需求、文档中明确描述的开发标准，以及所有专业开发的软件都应具备的隐含特征的程度。

1983 年，在 IEEE std 729-1983 标准中，对软件质量给出了下列更为具体、完整的定义：

软件产品中所能满足用户给定需求的全部特性的总体能力；软件具有所期望的各种属性组合的程度；用户主观得出的软件是否满足其综合期望的程度；软件的组合特性满足用户预期需求的程度。

这一定义重点强调了软件的特性或特征，强调了这些特性或特征与软件需求的吻合程度以及对它们的综合评价值。一般而言，不同性质和用途的软件，会有不同的特征集合和质量要求。如实时控制软件和大型联机事务处理软件对于软件的可靠性要求很高。而常规的办公事务软件及管理信息系统对于易用性和可移植性要求则比较高。

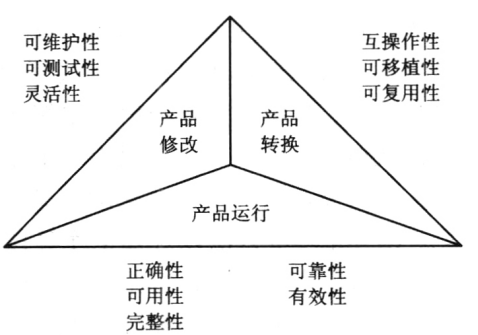


图 1.2　McCall 的软件质量模型

软件质量通常采用质量模型来建立用户视角和开发者视角的软件质量特性间的关系，如 Bohem 的质量模型、McCall 的质量模型以及 ISO 的质量模型。图 1.2 为 McCall 的质量模型。

McCall 的模型中给出了 11 个质量要素，如表 1.1 所示。11 个质量要素分为三类，对应软件的运行特性、修改特性和转换特性。软件的运行特性包括正确性、可靠性、有效性、

完整性和可用性；修改特性是指软件承受修改的能力，包括可维护性、灵活性和可测试性；软件的转换特性包括可移植性、可复用性和互操作性。

表 1.1 McCall 的软件质量要素准则

要素	描 述
正确性	指软件达到或满足需求规格说明及完成用户目标的程度。这里需要强调的是，用户需求不仅包括显性陈述的需求内容，还应包括隐含的需求
可靠性	指软件在指定条件下和特定时间段内维持其正常性能水准的能力的程度
完整性	系统不遗漏和丢失应实现的功能，完全实现需求规格说明所定义目标的程度
可用性	指用户为使用一个软件产品所付出的学习努力以及其他代价的程度
有效性	指在规定的条件下，软件功能与所占用资源之间的比值关系
可维护性	当发现错误、运行环境改变或客户需求改变时，程序可被修改的难易程度
可测试性	程序接受测试达到预定目标的能力
灵活性	程序易于修改、扩展、适应的能力
可移植性	指将软件从一种环境移植到另一种环境的难易程度
可复用性	软件在其他系统中可再次使用的程度或范围
互操作性	与其他系统交换信息和使用信息的能力

McCall 的模型中给出了一组较易度量的软件质量评价准则，共 20 种。评价准则能够比较完整、准确地描述质量，且比较容易量化和度量。表 1.2 列出了对它们的描述。

表 1.2 McCall 的软件质量评价准则

准 则	描 述
可审查性	检查软件需求、规格说明、标准、过程、指令、代码及合同是否一致的难易程度
准确性	软件计算和控制的精度，表示为相对误差的函数，函数值越大精确度越高
完整性	系统不遗漏和丢失应实现的功能，完全实现需求规格说明所定义目标的程度
简明性	程序源代码的紧凑性
一致性	软件系统使用一致的概念、符号、术语、接口、规范的程度
数据通用性	在程序中使用标准的数据结构和类型的程度
容错性	系统在异常条件下继续工作的能力
执行效率	程序运行的效率，通常以响应时间，内存占用率等来度量
可扩充性	能够对系统结构、数据结构进行扩充的程度
通用性	程序部件潜在的应用范围的广泛性
硬件独立性	软件与支撑其运行的硬件系统不相关的程度
检测性	程序运行发生错误时，检测标识错误的程度
模块化	程序中逻辑上相对独立、具有良好接口的模块化程度，模块内聚度、耦合度是常见的度量指标
可操作性	操作一个软件的容易程度
安全性	系统控制和保护程序或数据不受破坏的能力
自文档化	源代码提供有意义文档的程度
简单性	理解程序的难易程度
软件独立性	程序与非标准的程序设计语言、操作系统以及其他软件环境约束无关的程度
可追踪性	根据软件需求对软件设计、程序进行正向追踪，或根据程序、软件设计对需求进行逆向追踪的能力
易培训性	软件支持新用户使用该系统的能力

软件质量要素 F_j 的评价可用下列公式计算：

$$F_j = \sum_{k=1}^{l} C_{jk} M_k \qquad j = 1, 2, \cdots, 11$$

其中，M_k 是软件质量要素 F_j 对第 k 种评价标准的测量值，C_{jk} 是相应的加权系数。

由于 McCall 定义的评价准则多数没有客观的度量方法，因此只能凭主观印象为评价准则定值。McCall 将评价准则分为 0～10 级，0 级最低，10 级最高。加权系数 C_{jk} 满足 $\sum_{k=1}^{l} C_{jk} = 1$，其中 $C_{jk} \geqslant 0$，当质量要素 F_j 与 k 项评价准则无关时，$C_{jk} = 0$，l 表示评价准则的项数，对于 McCall 的评价准则，$l = 20$。

McCall 等人提出的软件质量度量模型、软件质量要素和评价准则为软件质量管理奠定了基础。近年来，国内外许多软件工程组织和专家在软件质量要素和评价准则的选取度量方面做了大量工作，部分结果已用于软件开发过程的质量控制和软件产品的质量度量。

1.2　软件工程概述

1.2.1　软件危机

在 20 世纪 60 年代末软件工程概念提出之前，由于计算机硬件资源的限制(运算速度、内存容量)，程序员不得不运用个人的编程技巧来提高代码效率，使得程序的可理解性较差。规模较小的程序尚能应付测试和运行维护，然而随着计算机硬件和软件技术的发展，软件的应用领域越来越广，软件的规模和复杂度也在日益增大。一个软件项目可能会有几万、几十万甚至上百万行代码，项目的完成也需要多人的合作，软件的可读性、人员之间的协调、项目管理的问题变得突出，软件的质量和生产率开始出现问题，软件和软件开发技术未能很好地满足软件产品的开发要求，软件危机(Software Crisis)现象爆发。软件危机主要表现在以下几方面：

(1) 软件开发无计划性。由于缺乏软件开发的经验和有关软件开发数据的积累，使得开发计划难以制定。而且往往出现计划的执行与计划安排有较大差距，致使开发进度难以保证，经费突破预算等现象频频出现。

(2) 软件需求不充分。软件需求是对软件所要实现的目标、功能及其相关限制的定义，是软件开发前期的重要工作。需求不能充分、准确地描述将会为后来的软件设计和开发带来隐患，造成后期开发活动中问题的集中暴露。需求不充分现象的形成主要在于用户对软件需求的描述可能不够准确，存在遗漏和二义性。另外软件人员对需求的理解与用户的期望也会存在差异。

(3) 软件开发过程无规范。软件开发过程没有统一的、公认的方法论和规范指导，不重视过程记录，使得设计和实现过程的文档不完整，忽视开发人员之间的接口问题，使得软件的开发效率和质量无法得到保证，所开发的系统也难以维护。

(4) 软件产品无评测手段。没有有效的方法和手段对软件进行全面的验证和测试，致使

所交付的软件往往存在较多的缺陷和质量问题，给软件的应用带来隐患。

综上所述，软件危机一直存在于软件发展的进程中，软件开发面临众多挑战。软件工程则是人们期望采用工程的方法进行软件的生产，以应对软件危机，目前软件工程的研究在不断发展，其原理、方法和工具已在软件开发中发挥巨大作用。

1.2.2 软件工程

1968年，在讨论有关软件危机的NATO(北大西洋公约组织)会议上提出了“软件工程”概念。软件工程是一门工程学科，它涉及软件开发过程的各个方面。它包括运用计算机、工程、数学等学科的理论和方法解决软件问题，同时也涉及到软件质量的保证、项目管理等方法和技术，还包括支持软件过程开展的工具。一般来说，软件工程采用系统和组织的方法以达到最有效的生产高质量软件的目的，从而经济地获得可靠的、可以在实际机器上高效运行的软件。工程不仅仅是一个学科或一个知识体系，它是一个动词，一个行为动词，一个解决问题的方法。

IEEE关于软件工程的定义为：

软件工程是：① 将系统性的、规范化的、可定量的方法应用于软件的开发、运行和维护，即将工程化应用到软件上；② 对①中所述方法的研究。

软件工程具有三要素：方法(Methods)、工具(Tools)和过程(Procedures)。

软件工程方法可提供如何构造软件的技术。软件工程方法覆盖面很广，包括需求分析、设计建模、编程、测试和其他的支持。软件工程方法依赖于一组基本原则，如Demarco(1978)和Jacobson(1983)提出的结构化的方法，Booch(1994)和Rumbaugh(1991)建议的面向对象的方法。软件方法包括一组对需求和设计、算法过程、编码、测试和维护的定义元素等。软件工程方法贯穿于软件开发过程，起着核心的作用。表1.3列出了一些软件工程方法元素。

表1.3 软件工程方法元素

方法元素	描 述	举 例
系统模型描述	用特定的符号定义描述被开发的系统模型	对象模型，数据流模型
规则	运用于系统模型的约束规则	系统模型中的每个实体只能有一个唯一名
建议	好的建议利于构造一个良好的系统模型	一个类最好不要包含多于7个子类
过程指导	对系统开发过程活动的描述	在定义一个对象的操作前应为对象的属性建立文档

软件工程工具提供自动化或半自动化的支持，包括模型建立、代码生成、程序静态分析、软件测试、过程管理等。工具的集合称为计算机辅助软件工程(Computer—Aided Software Engineering)系统，简称CASE。CASE把软件、硬件、软件工程数据库(包括分析、设计、编码和测试重要信息的数据结构)组成一个软件工程环境(Environment)。例如IBM Rational系列软件开发工具包括了需求定义和管理、模型构建、代码分析、功能和性能测试、项目管理及集成开发平台等工具。工具可以有效地提高软件开发的效率和质量。

软件工程过程是用合适的方法、语言和工具由软件工程师进行软件活动的集合。软件工程过程定义了一个框架，是软件项目管理控制的基础。软件工程过程建立了一个环境以便于技术方法的采用、可交付产品(文档、报告以及格式等)的产生，并帮助确保质量和变更