

软件

测试技术

林连进 谢怀民 主编



北京理工大学出版社

BEIJING INSTITUTE OF TECHNOLOGY PRESS

高等职业教育“十三五”规划教材

软件测试技术

林连进 谢怀民 主 编
林土水 副主编

 **北京理工大学出版社**
BEIJING INSTITUTE OF TECHNOLOGY PRESS

图书在版编目 (CIP) 数据

软件测试技术/林连进, 谢怀民主编. —北京: 北京理工大学出版社, 2018. 8
(2018. 9 重印)

ISBN 978 - 7 - 5682 - 6057 - 2

I. ①软… II. ①林… ②谢… III. ①软件 - 测试 IV. ①TP311. 55

中国版本图书馆 CIP 数据核字 (2018) 第 182673 号

出版发行 / 北京理工大学出版社有限责任公司

社 址 / 北京市海淀区中关村南大街 5 号

邮 编 / 100081

电 话 / (010) 68914775 (总编室)

(010) 82562903 (教材售后服务热线)

(010) 68948351 (其他图书服务热线)

网 址 / <http://www.bitpress.com.cn>

经 销 / 全国各地新华书店

印 刷 /

开 本 / 787 毫米 × 1092 毫米 1/16

印 张 / 21

字 数 / 495 千字

版 次 / 2018 年 8 月第 1 版 2018 年 9 月第 2 次印刷

定 价 / 53.00 元

责任编辑 / 钟 博

文案编辑 / 钟 博

责任校对 / 周瑞红

责任印制 / 施胜娟

编 委 会

主 任：俞发仁

副主任：王秋宏 黄可明 钟开华

委 员：	林土水	林连进	谢怀民	陈圣立	陈有富
	江 舒	江文通	韩领哲	林斯潇	李 思
	刘良清	周友禄	郑思思	陈雯艳	江满连
	卢 山	叶挺华	林志宏	曾娇艳	吴 静
	林 滨	郑志建	胡长生	李 楠	陈二微
	杨颖颖	陈秀丽	焦 博	王婷婷	

前 言

20 世纪 40 年代，当 Grace Hopper 中尉第一次在“事件记录本”中把引起 MARK II 计算机死机的飞蛾注明为“第一个发现虫子的实例”后，人们便将计算机和软件的错误戏称为虫子（bug）或臭虫，用此描述再恰当不过。对于软件开发人员和使用者来说，软件的缺陷就像自然界中的臭虫一样，是一场噩梦，轻则给用户带来不便，如软件界面不一致；重则造成重大生命财产的损失，如 1996 年阿丽亚娜 5 型火箭第一次鉴定发射的失败以及第一次海湾战争中爱国者导弹在沙特阿拉伯的多哈误炸 28 名美国士兵的事件。要找出软件中的问题，软件测试是唯一的手段。

同时，随着软件规模和复杂性的大幅度提升，软件质量可靠性的问题变得日益突出。软件测试是保证软件质量的关键技术之一，也是软件开发过程中的一个重要环节，其理论知识和技术工具都在不断革新。

本书较为全面地介绍了当前测试领域的专业知识，追溯了软件测试的发展史，反映了当前最新的软件测试理论、标准、技术和工具，展望了软件测试的发展趋势。全书共分十章，分别是软件测试的目标、软件工程概况、软件测试基础概述、软件测试方法概述、软件测试方法和技术、软件接口测试、软件 UI 自动化测试、软件性能测试、软件安全测试、软件测试过程。

软件测试从软件工程中演化而来，并且还在不断地发展。在学习本书之前，需要一些先行知识作为本书的支撑，如掌握一门高级语言（Visual Basic 或 Java），数据库、数据结构以及软件工程的基本理论知识等。

本书介绍了软件测试的基本理论和当前流行的一些软件测试工具的应用，内容精练，文字简洁，结构合理，综合性强，定位明确，面向初、中级读者，由“入门”起步，侧重“提高”，特别适合作为相关专业软件测试的教材或教学参考书，也可供从事计算机应用开发的各类技术人员参考，或用作全国计算机软件测评师考试、软件技术资格与水平考试的培训资料。

本书由网龙网络有限公司、福州软件职业技术学院联合编写，网龙网络有限公司高级工程师林连进、福州软件职业技术学院软件工程系主任谢怀民担任主编，福州软件职业技术学院林土水担任副主编。

在本书的写作过程中，北京理工大学出版社的编辑对写作大纲、写作风格等提出了很多宝贵的意见。本书在写作过程中参阅了大量国内外的专著、教材、论文、报告及网上的资料，由于篇幅所限，未能一一列出。在此，向各位作者表示敬意和衷心的感谢。

由于作者水平有限，本书难免有不足之处，诚恳期待读者的批评指正，以使本书日臻完善。

编 者

目 录

第一章 软件测试的目标	1
1.1 测试目标的定义	1
1.2 测试目标的界定	1
1.3 目标的拆解与分析	2
1.4 分析模型	3
1.5 确保目标达成的策略	5
1.6 确保目标达成的体系	6
第一章习题	7
第二章 软件工程概况	8
2.1 软件工程的发展历史	8
2.2 现代软件工程的定义	9
2.3 现代软件工程的基本原理	10
2.4 现代软件工程框架	11
2.5 现代软件工程方法学	11
2.6 软件的生命周期	16
2.7 软件生命周期模型	19
2.8 软件工程的目标和原则	26
第二章习题	27
第三章 软件测试基础概述	28
3.1 软件测试的历史	28
3.2 软件测试涉及的关键问题	29
3.3 软件测试与软件质量保证	30
3.4 软件故障的定义及分类	31
3.5 软件测试原则	34
3.6 停止测试的标准	38
3.7 软件测试人员的要求	41
第三章习题	43
第四章 软件测试方法概述	45
4.1 基于生命周期的软件测试	46
4.2 黑盒测试与白盒测试	52
4.3 静态测试与动态测试	55
4.4 验证测试与确认测试	57



第四章习题	58
第五章 软件测试的方法和技术	59
5.1 软件功能测试的定义	59
5.2 黑盒测试方法——等价类划分法	59
5.3 黑盒测试方法——边界值分析法	61
5.4 黑盒测试方法——决策表法	62
5.5 黑盒测试方法——因果图法	63
5.6 黑盒测试方法——场景法	65
5.7 黑盒测试方法——错误推测法	66
5.8 白盒测试	66
第五章习题	68
第六章 软件接口测试	70
6.1 接口测试的概念	70
6.2 HTTP 协议基本知识	71
6.3 RESTful 接口	87
6.4 接口测试流程	92
6.5 接口测试用例设计	98
6.6 接口测试质量评估标准	101
6.7 接口测试工具	101
6.8 接口测试自动化	107
第六章习题	111
第七章 软件 UI 自动化测试	113
7.1 UI 自动化测试介绍	113
7.2 PC 端 UI 自动化测试	114
7.3 Web 自动化测试	115
7.4 移动端 UI 自动化框架	127
7.5 脚本编写规范	138
第七章习题	140
第八章 软件性能测试	141
8.1 什么是性能测试	141
8.2 性能测试流程体系	142
8.3 性能测试技术体系	144
8.4 性能测试工具介绍 (LoadRunner)	149
8.5 性能监控分析工具介绍	176
8.6 案例分享：性能测试与分析	193
第八章习题	198
第九章 软件安全测试	199
9.1 如何做好软件安全测试	199



9.2 名词术语	200
9.3 常见安全测试工具介绍	203
9.4 常见安全测试案例分析	272
第九章习题	302
第十章 软件测试过程	303
10.1 过程模型	303
10.2 软件测试过程的关键活动	306
10.3 软件测试计划	307
10.4 测试用例设计	308
10.5 软件测试执行	309
10.6 缺陷管理	310
10.7 测试报告	314
第十章习题	319
参考文献	320

第一章

软件测试的目标

在工作上接受任何任务，首先要思考其目的是什么，有了对任务目标的充分认识以后，才可以对后续的核心问题或者要点进行分析，进而设计应对方案。

软件测试是一项技术，是一些问题的解决方案。本书首先介绍软件测试的目标，然后解释为了达成目标需要解决哪些核心问题，进而介绍不同问题对应哪些不同的软件测试技术。

1.1 测试目标的定义

软件测试的核心目标是保障软件质量。

国际标准化组织（ISO）在质量特性国际标准 ISO/IEC 9126 中将软件质量定义为反应软件产品满足规定需求和潜在需求的能力的特征和特性总和。

软件质量就是描述计算机优秀程度的特性的组合。也就是为了满足软件的各项精确定义的功能、性能要求，符合文档化的开发标准，需要相应给出或设计一些质量特性及其组合，要得到高质量的软件产品，就必须使这些质量特性得到满足。

按照 ANSI/IEEE Std 1061 - 1992 中的标准，软件质量是与软件产品满足规定的和隐藏的需求的能力有关的特征或特性的全体，包括：

- （1）软件产品中能满足用户给定需求的全部特性集合；
- （2）软件具有各种属性组合的程度；
- （3）用户主观得出的软件是否满足其综合期望的程度；
- （4）决定所用软件在使用中将满足其综合期望程度的合成特性。

综上所述，软件质量是许多质量属性的综合体现，各种质量的属性反映了软件质量的方方面面。人们通过改善软件的各种质量属性，提高软件的整体质量。

1.2 测试目标的界定

什么是“好的品质”？世界上没有绝对好的质量，只有相对合适的质量标准。

对于某些初创软件开发团队来说，获得投资人认可的质量，不一定是最优的质量，但却是现阶段最合适的质量标准。因为若没有达到这个标准，难以保障他们的团队“活下去”，若超过这个标准，则需要消耗太多的人力、物力，以及最宝贵的时间，即不是他们想要的了。

对于互联网行业的某些巨头来说，软件产品的质量重点体现在性能承载能力，而非功



能。因为其某些产品形态和商业模式决定了他们对高并发的诉求，远胜于他们对产品一般功能完整或多样性的要求。

对于某些机要部门或单位的系统来说，保障安全准确才是第一要务，一切以“稳”为主。

所以，对质量诉求的不同导致各个不同的机构，甚至同一个机构内部，软件质量的标准也会不一样。这是在尝试认清自己的目标测试软件系统的质量目标时首要需要明确的内容。

1.3 目标的拆解与分析

上面讨论了什么是软件测试的目标，应当如何从标准定义测试目标，以及基于不同的团队实际场景来理解该目标。

下面讨论怎样对目标进行分析和拆解。

这里介绍一个通用的分析事物的方法论，就是首先查看是否有专业的模型可以套用分析，以此确保一开始就能够着眼目标事物的全貌，这对于已经相对成型的专业领域尤为有效。

在质量领域，市面上有不同的质量模型，可以帮助对目标软件进行各类属性上的拆分。同时，可以考虑借助不同的质量模型建立更匹配的全局感。

传统的质量模型往往从产品本身的属性出发分析各方面的标准和要求，而本书认为在深入分析各类属性之前，应该有两个更重要的维度走在传统质量模型的前面，即产品和行业。

1. 行业维度

行业维度主要有两点内容。

1) 用户

对产品的质量进行分析，先要明确为什么样的人/人群服务，不同的人/人群关心的重点不一样。

这里举一个案例：

某个手机游戏产品，一开始的运营定位为三四线城市的玩家，非头部玩家。这一定位可能会影响哪些质量属性呢？

- (1) 用户使用时长；
- (2) 用户使用的机型差异；
- (3) 服务器的设计模式与部署方案；
- (4) ……

上面的每一个点所针对的不仅是研发的上游，产品开发人员需要对这类信息进行具有针对性的分析。

2) 竞品

对于某个产品，如果市场上没有细分品类竞品，那么很可能出现下面两种情况：①这是一个绝佳的市场和时机，面对蓝海有足够的议价能力；②原先驻扎在该领域的竞品都没能



“活”下来。

无论是上面的哪一种情况，都需要整个研发团队掌握快速试错的能力。与质量相关的关注点，就应该集中在一两个用户最关注的核心交付价值上。

相反，如果该领域内已经驻扎着明确的竞品，那么在进行质量分析的时候，就不光要看到自己产品的成长和进步，也要同时关注竞品具有哪些质量属性，进行阶段性的拆解比对，甚至将其当作项目质量目标的基准线。

2. 时间维度

时间维度的分析，要求能够明确产品的大目标、阶段性里程碑目标，以及对于产品不同分支的重点要求，即需要将产品各模块里程碑诉求和时间对应起来，建立一个时间点和关键质量的匹配表，见表 1-1。

表 1-1 时间点和关键质量的匹配表

时间点	里程碑	质量指标
Yyyy - mm	里程碑 A: 描述……	• 指标 a1 • 指标 a2 •
Yyyy - mm	里程碑 B: 描述……	• 指标 b1 • 指标 b2 •
.....		

1.4 分析模型

在明确了上一节所介绍的分析维度之后，就可以进一步通过质量模型来分析自身产品的质量目标了。

质量模型是产品质量评价体系的基石。质量模型决定哪些质量特性将被纳入软件产品评估。

软件质量是软件系统满足各类利益相关者诉求的体现。质量模型则对这些诉求（功能、性能、安全性、可维护性等）进行分门别类的评估。

质量模型在软件行业上出现过多个版本，本书仅介绍国际上最新的行业标准以供读者学习参考，即 ISO/IEC 25010 - 2011 模型规范。

ISO/IEC 25010 模型规范包含 8 个质量维度。

1. 功能维度

这一维度体现产品或系统在一定条件下满足用户需求的程度。它由下述子维度组成：



- (1) 功能的完整性：功能涵盖了所有指定的任务和用户目标的程度。
- (2) 功能的准确性：产品提供正确结果的准确程度。
- (3) 功能适用性：帮助指定任务和目标达成的有效程度。

2. 性能维度

这一维度代表了性能在规定的资源条件下的使用情况。它由下述子维度组成：

- (1) 时间：代表系统某功能响应时间和吞吐率满足需求的程度。
- (2) 资源利用率：系统使用某类资源的类型和数量与需求匹配的程度。
- (3) 容量：系统某属性支持的最大值满足要求的程度。

3. 兼容性

兼容性指系统和其他系统或组件在一个软/硬件环境下交换信息并执行程序的程度。它由下述子维度组成：

- (1) 共存：系统与其他产品共享一个环境和资源时，能有效地执行其要求功能，并没有对其他产品造成不良影响。
- (2) 交互性：两个或两个以上的系统、产品或组件可以有效交换并使用信息。

4. 可用性

可用性指指定用户有效地、有效率地，并且满意地使用系统达成期望目标的程度。它由下述子维度组成：

- (1) 可辨识：用户可以顺利识别产品或系统是否适合其需求。
- (2) 易学性：在指定的上下文环境下，令用户可以有效地、有效率地、无风险地使用产品或系统。
- (3) 可操作性：产品或系统易于操作和控制的程度。
- (4) 操作规范性：系统避免用户误操作的能力。
- (5) 美观：用户交互界面带来美好观感的程度。
- (6) 可达性：使产品或系统在一定上下文环境下，可以被人们最大限度地使用，以达到特定目标。

5. 可靠性

可靠性是系统、产品或组件在特定时期执行特定功能的能力维度。它由下述子维度组成：

- (1) 成熟性：系统、产品或组件在正常操作下的可靠性。
- (2) 可用性：系统、产品或组件按需求可达。
- (3) 容错性：系统、产品或组件在故障或错误发生时的运行能力。
- (4) 恢复力：在发生中断或失败的时候，一个产品或系统可以恢复数据，并重建所需的系统状态的程度。



6. 安全性

安全性是他人或系统在相应的授权下，访问该产品或系统保护的信息和数据的能力维度。它由下述子维度组成：

- (1) 机密性：一个产品或系统确保数据只能被授权者访问。
- (2) 完善性：系统、产品或组件可以防止未经授权的访问与修改程序数据的行为。
- (3) 不可否认性：行为或事件存在即可证明已经发生，不能被抵赖的程度。
- (4) 问责：行动可以追溯到唯一的实体。
- (5) 真实性：主题或资源实体可以被证明。

7. 可维护性

可维护性指产品或系统改进、纠正或适应环境变化和需求的有效性和效率的程度。它由下述子维度组成：

- (1) 模块化：系统或程序中模块的离散程度，其使得对一个组件的更改对其他组件的影响最小。
- (2) 可重用性：资源可以被多个系统利用的程度。
- (3) 可分析性：评估产品或系统对其一个或多个部件的预期变化的影响，或诊断产品的缺陷或故障原因，或识别要修改的部件的有效性，或有效率程度。
- (4) 可修改性：在不引入缺陷或降低现有产品质量的情况下有效和有效率地修改产品或系统的程度。
- (5) 可测试性：可以为系统、产品或组件建立有效性和效率的测试标准，并进行测试以确定这些标准是否已满足。

8. 可移植性

可移植性指系统、产品或组件可以从一个硬件、软件或其他操作系统或使用环境转移到另一个系统的有效性和效率的程度。它由下述子维度组成：

- (1) 适应性：产品或系统有效地、有效率地适应不同或不断变化的硬件、软件或其他操作系统或使用环境的程度。
- (2) 可安装性：产品或系统在特定环境下有效地、有效率地安装或卸载的能力。
- (3) 可置换性：在同一环境中，在统一的执行目的下，产品替代另一特定软件产品的程度。

1.5 确保目标达成的策略

在厘清目标以及达成目标的评估手段之后，需要明确以什么样的策略达成目标，即当明确战略规划之后，需要进一步研究战术问题。

在这里先讲一个故事：

中国古代有一个著名的神医，名叫扁鹊。传说扁鹊有两个哥哥，他们也全是郎中。人们



问扁鹊：“你们兄弟三人谁的医术最高？”

扁鹊回答说：“我常用猛药给病危者医治，偶尔有些病危者被我救活，于是我的医术远近闻名并成了名医。我二哥通常在人们刚刚生病的时候马上就治愈他们，临近村庄的人说他是好郎中。我大哥不外出治病，他深知人们生病的原因，所以能够预防村里人生病，他的医术只有我们村里人才知道。其实我的医术不如二哥，二哥的医术不如大哥。”

从这里看出，要想身体好，少生病，有三种办法：

第一是早预防，不让疾病产生，这就是扁鹊大哥的方法。

第二是早发现，早治疗，这就是扁鹊二哥的方法。

第三是早抢救，死马当活马医。这是没有办法的办法，万不得已而为之。这就是扁鹊经常做的事。

这就是古话说的：“上工治未病，中工治欲病，下工治已病”。这里的“上工”指的是良医，以此类推。

首先，这里强调的是对事物发展的预见性，这也是管理中的高阶能力。能够预见各类风险，给出对应的预防措施，将风险消灭于无形，这是质量管理的高阶方法。

其次，在投入过程中，通过分析判断，发现出现偏差的时刻，再去应对响应，这往往是因为项目已经有了自己的轨道和节奏，再想改变，就应重点考验影响力了。而在故障已发生，质量受损的情况下再去应对，这就是低阶的执行力了。

通过从上工到下工的演变，可以清晰地看到在修复/恢复成本加大的同时，管理技能的各个修炼阶段，这就体现了战术的优劣。

在许多传统的概念里，测试就是不断地检验、检测不同指标的达成程度。实际上，测试是一个与时间赛跑的活动。如何在整个研发阶段跟上质量演变的步伐则是下一章的内容。

1.6 确保目标达成的体系

以战争为例，打赢一场战争，获得每一场胜利，是多方面配合的结果，需要情报、后勤、通信、医疗等多个分支协同合作。

质量也是一样，并不是测试人员单枪匹马就能达成目标，需要调动组织结构、流程保证、技术能力等各方面共同作用。

但现实是，很难从一开始就把上述所有方面都调整到“完美状态”。那么就需要根据实际情况和项目的演变阶段来不断调整策略，调动所能调动的资源以达成目标。

仍以战争为例，当我们兵力充足的时候，可以考虑包围战；而当兵力薄弱的时候，就要考虑集中兵力单点突围；当兵力少而精的时候，可以考虑特种兵式的外科手术般的战斗……

希望读者铭记的口诀是“看全局，抓重点”。

下面介绍一个测试能力成熟度模型（Testing Maturity Model, TMM），仅作为在构建质量体系，审视各阶段时的参考，见表 1-2。



表 1-2 测试成熟度模型

级别	描 述
1. 初始	在这个级别，一个组织正在使用应急的方法进行测试，所以结果是不可重复的，没有质量标准
2. 定义	这个级别的测试被定义为一个过程，因此可能有测试策略、测试计划、测试用例。基于需求，测试是在产品完成之前才开始的，因此测试的目的是比较产品和需求
3. 集成	在这个级别，测试被集成到软件生命周期模型，例如测试的需求是基于风险管理的，测试是从开发阶段便开始独立实施的
4. 管理和评价	在这个级别，管理和测试活动在产品生命周期的各个阶段进行，包括对需求和设计的评审。组织（内部和外部）所有产品的质量标准都是一致的
5. 优化	在这个级别，测试成了测试并改进每一轮的迭代。这通常通过工具实现，并且通过在全生命周期防止缺陷产生而不是缺陷检测（零缺陷）达成目标

从上面的 5 个层次中，可以提取出一个完善的体系所必须具备的一些条件，如场景分析、计划、标准、用例、流程支持、各色测试工具、组织内意识认同等。而最核心的是对于产品质量的匠师精神，以及对于其上任何维度的持续改进的激情和动力。

第一章习题

1. 软件测试的核心目标是_____。
2. 如何进行目标的拆解和分析？
3. 请举出一个软件测试模型，并说明针对某个目标测试软件，它的重点在哪里。
4. 质量保障策略是什么？
5. 针对某个质量目标，应该构建哪些保障手段？其中现阶段的重点是什么？

第二章

软件工程概况

2.1 软件工程的发展历史

软件工程（Software Engineering）是在处理 20 世纪 60 年代末所出现的“软件危机”的过程中逐渐形成与发展的。在计算机发展的早期（20 世纪 60 年代中期以前），通用硬件相当普遍，而软件却是为每一个具体应用专门开发的。这时的软件通常是规模较小的程序，编写者和使用者往往是同一个人或同一组人。这种个体化的软件环境，使得软件设计通常是在人们头脑中进行的一个隐含过程，除了代码之外，没有其他文档资料保存下来。

软件工程是一门研究如何用系统化、规范化、数量化等工程原则和方法进行软件开发和维护的学科。软件工程包括两方面内容：软件开发技术和软件项目管理。软件开发技术包括软件开发方法学、软件工具和软件工程环境。软件项目管理包括软件度量、项目估算、进度控制、人员组织、配置管理、项目计划等。

为了迎接软件危机的挑战，人们进行了不懈的努力。这些努力大致上是沿着两个方向同时进行的。

一是从管理的角度出发，希望实现软件开发过程的工程化。这方面最为著名的成果就是提出了大家都很熟悉的“瀑布式”生命周期模型，也称为传统的软件开发模式。它是在 20 世纪 60 年代末“软件危机”后出现的第一个生命周期模型。“瀑布式”软件开发的模式大体分为以下几个阶段：需求分析、设计、编码、测试、维护。

后来，又有人针对该模型的不足，提出了快速原型法、螺旋模型、喷泉模型等对“瀑布式”生命周期模型进行补充。这方面的努力还使人们认识到了文档的标准以及开发者之间、开发者与用户之间的交流方式的重要性。一些重要文档格式的标准被确定下来，包括变量、符号的命名规则以及源代码的规范式。

软件工程发展的第二个方向，侧重于对软件开发过程中分析、设计的方法的研究。这方面的重要成果就是在 20 世纪 70 年代风靡一时的结构化开发方法，即 PO（面向过程的开发或结构化方法）以及结构化的分析、设计和相应的测试方法。

软件工程的目的是研制、开发与生产具有良好的质量和费用合算的软件产品。费用合算是指软件开发运行的整个开销能满足用户要求的程度，软件质量是指该软件满足明确的和隐含的需求的能力的有关特征和特性的综合。软件质量可用六个特性来评价，即功能性、可靠性、易用性、高效性、维护性、易移植性。



2.2 现代软件工程的定义

早期的软件开发仅考虑人的因素，传统的软件工程强调物性的规律，现代软件工程最根本的因素就是人跟物的关系，即人和机器（工具、自动化）在不同层次的不断循环发展的关系。

面向对象分析（OOA）、面向对象设计（OOD）的出现使传统开发方法发生了翻天覆地的变化。随之而来的是面向对象的建模语言（以 UML 为代表）、软件复用、基于组件的软件开发等新方法和新领域。

软件工程一直以来都缺乏一个统一的定义，很多学者、组织机构都分别给出了自己的定义。Boehm 指出，软件工程就是运用现代科学技术知识来设计并构造计算机程序及开发、运行和维护这些程序所必需的相关文件资料。IEEE 在软件工程技术汇编中将软件工程定义为：将系统化的、严格约束的、可量化的方法应用于软件的开发、运行和维护，即将工程化应用于软件。Fritz Bauer 在 NATO 会议上给出的定义是，建立并使用完善的工程化原则，以经济的手段获得能在实际机器上有效运行的可靠软件的一系列方法。《计算机科学技术百科全书》中对软件工程的定义为：软件工程是应用计算机科学、数学及管理科学等原理开发软件的工程。软件工程借鉴传统工程的原则、方法，以提高质量、降低成本。其中，计算机科学、数学用于构建模型和算法，工程科学用于制定规范、设计范型、评估成本及确定权衡，管理科学用于计算、资源、质量、成本等的管理。

目前比较认可的一种定义认为：软件工程是研究和应用如何以系统性的、规范化的、可量化的过程化方法去开发和维护软件，以及如何把经过实践考验而证明正确的管理技术和当前能够得到的最好的技术方法结合起来。

虽然人们对软件工程的定义不尽相同，但是人们普遍认为软件工程具有以下本职特性：

（1）软件工程关注大型程序的构造。通常把一个人在较短时间内写出的程序称为小型程序，把多人合作，用时半年以上写出的程序称为大型程序。现在的软件开发项目的通用构造为包含若干个相关程序的“系统”。

（2）软件工程的中心课题是控制复杂性。软件所解决的问题十分复杂，所以通常把问题分解成若干个可以理解的小部分，而且各部分之间保持简单的通信关系。用这种方法虽然不能降低问题的整体复杂性，但是却可使它变得易于管理。

（3）软件经常变化。为了使软件不被很快淘汰，必须让其随着所模拟的现实世界一起变化。因此，在软件系统交付使用后仍然需要耗费成本，而且在开发过程中必须考虑软件将来可能的变化。

（4）开发软件的效率非常重要。随着社会的进步，社会对新应用系统的需求越来越大，超过了人力资源所能提供的限度，软件供不应求的现象日益严重。因此，提高软件开发的效率非常重要。

（5）开发团队和谐地合作是软件开发的关键。软件处理的问题十分庞大，必须多人协同工作才能解决这类问题。因此，只有开发团队同心协力、和谐地合作才能开发出优质的软件。