“高等学校本科计算机类专业应用型人才培养研究”项目规划教材

# 软件工程

## Software Engineering

舒　坚　陈斌全　主编

高等教育出版社

“高等学校本科计算机类专业应用型人才培养研究”项目规划教材

# 软件工程

## Ruanjian Gongcheng

舒　坚　陈斌全　主编
张恒锋　杨丰玉　樊　鑫　编著

高等教育出版社·北京

内容提要

本书由浅入深、系统地介绍了软件工程的基本概念、基本原理、软件开发方法和技术、软件测试与维护、软件项目管理与质量保证，重点介绍了面向对象分析与面向对象设计。配合知识点的介绍，全书各章有简洁的举例，并以一个规模和难度适中的项目为例贯穿书中的主要章节，将软件工程的概念、理论和技术融入实践当中，加深读者对软件工程知识的认识和理解。同时，在每章后有与之对应的习题，供读者复习巩固。

本书可作为高等院校软件工程、计算机及相关专业软件工程课程的教材或教学参考书，也可供程序员、软件测试工程师、软件项目管理人员及其他专业技术人员参考。

**图书在版编目（CIP）数据**

软件工程 / 舒坚，陈斌全主编. --北京：高等教育出版社，2015.3

ISBN 978-7-04-041960-3

Ⅰ. ①软… Ⅱ. ①舒… ②陈… Ⅲ. ①软件工程-高等学校-教材 Ⅳ. ①TP311.5

中国版本图书馆 CIP 数据核字（2015）第 024040 号

| | | | |
|---|---|---|---|
| 策划编辑 倪文慧 | 责任编辑 倪文慧 | 封面设计 张 志 | 版式设计 杜微言 |
| 插图绘制 宗小梅 | 责任校对 刘 莉 | 责任印制 赵义民 | |

| | | | |
|---|---|---|---|
| 出版发行 | 高等教育出版社 | 咨询电话 | 400-810-0598 |
| 社 址 | 北京市西城区德外大街 4 号 | 网 址 | http://www.hep.edu.cn |
| 邮政编码 | 100120 | | http://www.hep.com.cn |
| 印 刷 | 北京东君印刷有限公司 | 网上订购 | http://www.landraco.com |
| 开 本 | 787mm×1092mm 1/16 | | http://www.landraco.com.cn |
| 印 张 | 17.5 | 版 次 | 2015 年 3 月第 1 版 |
| 字 数 | 380 千字 | 印 次 | 2015 年 3 月第 1 次印刷 |
| 购书热线 | 010-58581118 | 定 价 | 28.00 元 |



物 料 号 41960-00

# 前言

为了解决软件危机，北大西洋公约组织（NATO）于 1968 年首次提出了“软件工程”这一概念，软件开发开始了从“艺术”“技巧”和“个体行为”向“科学与工程”和“群体协同工作”转化的历程。“软件工程”如一线霞光，使软件发展步入了一个新的时代。经过 40 多年，特别是 20 世纪 90 年代以来的迅速发展，软件工程的研究和实践取得了很大的进步，经历了从结构化到面向对象的发展过程。

作者以教育部“卓越工程师教育培养计划”为指导，在参考了大量国内外教材与文献的基础上，结合多年从事本科生和研究生软件工程课程教学经验及软件项目开发的体会，针对应用型、工程型大学本科软件工程课程教学的特点编写了本书。

本书定位于应用型、工程型大学本科生，面向软件工程技术发展的需要，围绕结构化软件工程和面向对象软件工程方法，以软件生命周期为线索，以一个规模和难度适中的项目贯穿书中的主要章节，较为系统地介绍了计算机科学技术、软件工程等相关专业必需的软件工程知识。全书共 10 章。第 1 章介绍软件工程的相关概念、基本原理和软件工程知识体，第 2 章介绍软件开发中常用的开发模型，第 3 章介绍结构化软件工程开发方法，第 4、6、7 章介绍面向对象软件工程开发方法，第 5 章介绍可行性分析与项目计划制定，第 8 章介绍编码规范和软件测试方法，第 9 章介绍软件维护，第 10 章介绍软件项目管理与质量保证。

本书由南昌航空大学软件工程课程组完成，舒坚组织了本书的编写工作，并对全书进行了统稿。第 1、2、3 章由陈斌全编写，第 4、6、7 章由张恒锋编写，第 5、9、10 章由杨丰玉编写，第 8 章由樊鑫编写。

感谢南昌航空大学对本书出版给予的支持。

云南大学李彤教授认真地审阅了书稿，提出许多宝贵意见，指正了书稿中的不足之处，在此表示衷心的感谢。

由于时间仓促，水平有限，诚恳欢迎各位读者批评指正。

编　者

2014 年 12 月

# 目录

# 第1章 概论

软件工程是一门指导大中型计算机软件系统开发和维护的工程学科，它是在克服20世纪60年代末所出现的“软件危机”的过程中逐渐形成和发展的。本章介绍软件和软件工程的相关概念、发展历程，软件工程的基本原理、软件工程知识体等，使读者对软件工程的总体框架有初步的了解，便于以后各章的学习。

## 1.1 计算机软件

在计算机发展早期，软件被等同于程序，开发软件也就是编写程序。但是，随着软件应用的推广与规模的扩大，软件由程序发展成为软件产品，软件项目成为软件开发中的工程任务计量单位，软件作为工程化产品则被理解为程序、数据、文档等诸多要素的集合。应该说，软件工程是为适应软件的产业化发展需要而逐步发展起来的一门有关软件项目开发和维护的工程方法学。

### 1.1.1 计算机软件的定义

计算机软件是用户与硬件之间的接口，用户主要通过软件与计算机进行交流，软件是计算机系统设计的重要依据。为了方便用户，使计算机系统具有较高的总体效用，在设计计算机系统时，必须通盘考虑软件与硬件的结合，以及用户的要求和软件的要求。

计算机科学对计算机软件（Computer Software）的定义是：“软件是在计算机系统支持下，能够完成特定功能和性能的程序、数据和相关的文档”。程序是对计算任务的处理对象和处理规则的描述；数据指的是程序能够适当操作的信息；文档是为便于了解程序所需的阐明性资料。程序必须装入机器内部才能工作，文档一般是给人看的，不一定装入机器。

计算机软件可以形式化地表示如下：

软件=程序+数据（库）+文档+知识

计算机软件虽然由程序、数据和文档组成，但软件蕴含着“完成特定功能和性能”的知识和经验。软件的功能和性能除了取决于软件自身的质量外，还取决于软件描述的领域知识和经验。软件具有文化和技术双重属性。很难想象一个结构良好、不能解决实际问题的软件会有什么社会和经济价值。软件开发团队在注重软件开发方法的同时，更要注重求解问题的领域知识和经验，没有相关的领域知识和经验，软件就不能完成预定的功能和性能。因此，软件开发

团队必须善于与领域专家合作、妥善处理知识产权问题；同时，应有自己的特色和定位，长期在某一领域进行软件开发，掌握核心技术，提供优质服务，将会在竞争中形成明显的优势，如石油地震数据处理、中长期天气预报、银行业务处理、医疗器械控制等领域的软件。

### 1.1.2 计算机软件的特征

软件是计算机系统中的逻辑成分。相对于硬件的有形物理特性，软件是抽象的，具有无形性。例如，程序是操纵计算机工作的指令集合，但它并不能以一种特殊的物理形态独立存在，而必须通过它之外的物理存储介质，如磁盘、U 盘等，才能得以保存；当需要它工作时，又需要将它调入计算机的内部存储器中，通过计算机的中央处理器控制并进行相应运算才能工作。

软件的无形特性使得人们只能从软件之外去观察、认识软件。可以把计算机系统与人的大脑进行比较，计算机硬件如同人脑的生理构造，软件则如同基于人脑而产生的知识、思想等，具有与硬件完全不同的特征。软件的特征主要表现在以下几个方面。

（1）软件是硬件的灵魂，硬件是软件的基础

计算机硬件必须靠软件实现其功能，如果没有软件，硬件就好比一堆废铁，所以说软件是硬件的灵魂；同时，软件必须依赖于硬件，只有在特定的硬件环境上才能运行。

虽然“软件工厂”的概念已被引入，这并不是说硬件生产和软件开发是一回事，而是引用“软件工厂”这个概念促进软件开发中模块化设计、组件复用等意识的全面提升。

（2）软件是智慧和知识的结晶

软件是完全的智力产品，是通过开发人员的大脑活动创造的结果。软件属于高科技产品，软件产业是一种知识密集型产业。

软件的主要成本在于先期的开发人力。软件成为产品之后，其后期维护、服务成本也很高。而软件载体的制作成本很低，如 U 盘、光盘的复制是比较简单的，所以软件也易于成为盗版的主要目标。

（3）软件不会“磨损”，而是逐步完善

随着时间的推移，硬件构件会由于各种原因而受到不同程度的磨损，但软件不会。新的硬件故障率很低，经过长时间的使用，硬件会老化，故障率会越来越高；对软件来说则相反，隐藏的错误会使软件在其生命初期具有较高的故障率，随着使用的不断深入，所发现的问题会慢慢地被改正，其结果是软件越来越完善，故障率会越来越低。

硬件可用另外一个备用零件替换，但对于软件，不存在替换，而是通过开发补丁程序不断地解决适用性问题，或扩充其功能。一般来说，软件维护要比硬件维护复杂得多，而且软件的维护周期要长得多。正是通过不断地维护、改善或增加新功能，来提高软件系统的稳定性和可靠性。

### 1.1.3 计算机软件的分类

计算机系统往往需要许多不同种类的软件协同工作，软件工程人员也可能会承担各种不同种类的软件开发、维护任务。因此，可以从多个不同的角度来划分软件的类别。

（1）按软件功能划分

① 系统软件：计算机系统的必要成分，它跟计算机硬件紧密配合，以使计算机系统的各个部分协调、高效地工作。如操作系统、数据库管理系统等。

② 支撑软件：用于协助用户开发与维护系统的一些工具性软件。如程序编译器、源程序编辑器、错误检测程序、程序资源库等。

③ 应用软件：用于为最终用户提供数据服务、事务管理或工程计算的软件。如商业数据处理软件、工程设计制图软件、办公管理自动化软件、多媒体教学软件等。

（2）按软件工作方式划分

① 实时处理软件：能够及时进行数据采集、反馈和迅速处理数据的软件，其主要用于特殊设备的监控。如自动流水线监控程序、飞机自动导航程序。

② 分时处理软件：能够把计算机 CPU（Central Processing Unit，中央处理器）工作时间轮流分配给多项数据处理任务的软件。如多任务操作系统。

③ 交互式软件：能够实现人机对话的软件。这类软件往往通过一定的操作界面接收用户给出的信息，并由此进行数据操作；其在时间上没有严格的限定，但在操作上给予了用户很大的灵活性。如商业数据处理系统中的客户端程序。

④ 批处理软件：能够把一组作业按照作业输入顺序或作业优先级别进行排队处理，并以成批加工方式处理作业中的数据。如汇总报表打印程序。

（3）按软件规模划分

① 微型软件：一个人几天内即可完成的、源程序在 500 行语句以内的软件。这种软件主要供个人临时性使用，软件任务单一，操作简单，没有必要建立专门的软件系统文档。

② 小型软件：一个人半年之内即可完成的 2 000 行语句以内的软件。这种软件通常没有与其他软件的数据接口，主要用于用户企业内部的专项任务，大多由最终用户自主开发，软件使用周期短，但软件运行过程中产生的数据则可能在今后软件系统扩充中有一定的价值。考虑到软件系统今后有扩充的可能性，软件创建过程中应该提供必要的软件系统文档支持。

③ 中型软件：由一个项目小组在一年时间内完成的 5 万行源程序以内的软件系统。中型软件在项目实施过程中有了软件人员之间、软件人员与用户之间的通信，软件开发过程中需要进行资源调配。因此，软件开发过程中需要系统地采用软件工程方法，如项目计划、技术手册、用户手册等文档资源，以及技术审查制度等，都需要比较严格地建立起来。

④ 大型软件：由一个或多个项目小组在两年时间内完成的 5 万行源程序以上的软件系统。当多个项目小组共同参与开发时，需要对参加软件开发的软件工程人员实施二级管理，一

般将项目按功能子系统分配到各个项目小组，然后在项目小组内将具体任务分配到个人。由于项目周期较长，在项目任务完成过程中，人员调整往往不可避免，可能会出现对新手的培训和新手需要逐步熟悉工作环境的问题。对于这样大规模的软件，采用统一的标准，实行严格的审查是绝对必要的。由于软件的规模庞大以及问题的复杂性，往往在开发过程中会出现一些事先难以预料的事件，对此需要有一定的思想与工作准备。

（4）按软件服务对象划分

① 通用软件：由软件开发机构开发出来的直接提供给市场的软件。例如，通用财务软件、通用字处理软件、杀毒软件等。这类软件通常由软件开发机构自主进行市场调查与市场定位，并由此确定软件规格，大多通过一定的商业渠道进行软件销售。

② 定制软件：受某个或少数几个特定客户的委托，由一个或多个软件开发机构在合同的约束下开发出来的软件。如某专门设备的控制系统、某特定企业的业务管理系统、某智能大厦的监控与管理系统、某城市的交通监管系统等。这类软件通常由用户进行软件描述，并以此为基本依据确定软件规格。作为软件开发机构，则必须按照用户要求进行开发。

### 1.1.4 计算机软件的发展历程

计算机系统总是离不开软件的。早期的硬件、软件融于一体，为了使某台计算机设备能够完成某项工作，不得不给它专门配置程序。但是，随着计算机技术的快速发展和计算机应用领域的迅速拓宽，自20世纪60年代中期以来，软件需求迅速增长，软件数量急剧膨胀，于是软件从硬件设备中分离了出来，不仅成为独立的产品，并且逐渐发展成为一个专门的产业领域。

纵观软件的发展，可以发现软件生产有三个发展时代，即程序设计时代、程序系统时代和软件工程时代。

（1）程序设计时代（20世纪50年代）

20世纪50年代是软件发展的早期时代，计算机主要应用于科研机构的科学工程计算，软件则是为某种特定型号的计算机设备而专门配置的程序。这时的软件工作者以个体手工的方式制作软件，他们使用机器语言、汇编语言作为工具，直接面对机器编写程序代码。这时的硬件设备不仅价格昂贵，而且存储容量小、运行速度慢、运行可靠性差。尽管程序要完成的任务在今天看来是简单的，但由于受计算机硬件设备条件的限制，程序设计者也就不得不通过编程技巧来追求程序的运行效率。这个时期的程序大多是自用，程序的编写者往往就是使用者，软件还没有成为产品。

由于早期程序大多是为某个具体应用而专门编写的，程序任务单一，因此，对程序的设计也就仅仅体现在单个程序的算法上。早期程序还往往只能在某台专门的计算机上工作，很难将程序从一台设备移植到另一台设备。

（2）程序系统时代（20 世纪 60 年代）

20 世纪 60 年代，计算机技术迅速发展。在硬件技术上，由于半导体材料、集成电路的出现，计算机设备不仅在运行速度、可靠性和存储容量上有了显著的改善，而且价格也大大降低，这使得计算机得到了更加广泛的应用。例如，一些大型商业机构已经开始使用计算机进行商业数据处理。

在软件技术上，高级语言的诞生显著提高了程序编写的效率，这使得一些更大规模的具有综合功能的软件被开发出来。在该阶段出现了操作系统，它有效地改善了应用软件的工作环境，使应用软件具有了可移植性。在这个时期出现了“软件作坊”，伴随着“软件作坊”还产生了具有一定通用性的软件产品。

“软件作坊”已开始生产具有工业化特征的软件产品，并且此时的软件工作者已开始使用系统的方法设计、制作软件，而不是孤立地对待每个程序。但是，“软件作坊”是一种比较疏散的组织机构，这使得软件开发还不能形成工业流程。

这个时期的软件开发更多地依赖于个人创作。由于软件开发的主要内容仍然是程序编写，软件开发的核心问题仍是技术问题，于是用户的意图被忽视了，除了程序之外的其他文档、技术标准、软件在今后运行过程中的维护等问题，也往往被忽视了。

由于软件已经开始成为产品，但软件的生产方式却是与产品并不适宜的作坊创作方式。于是，随着软件规模的不断扩大，“软件危机”（Software Crisis）现象在这个时期最终爆发出来，即软件开发的质量、效率等均不能满足应用的需求。

（3）软件工程时代（20 世纪 70 年代起）

为了解决软件危机，北大西洋公约组织（North Atlantic Treaty Organization，NATO）于 1968 年首次提出了“软件工程”这一概念，软件开发开始了从“艺术”“技巧”和“个体行为”向“科学与工程”和“群体协同工作”转化的历程。“软件工程”如一线霞光，使软件发展步入了一个新的时代。

经过 40 多年，特别是 20 世纪 90 年代以来的迅速发展，软件工程的研究和实践取得了很大的进步，主要历程包括以下几个阶段。

① 20 世纪 60 年代末～20 世纪 70 年代中期，在一系列高级语言应用的基础上，出现了结构化程序设计技术，并开发了一些支持软件开发的工具。

② 20 世纪 70 年代中期～20 世纪 80 年代中期，计算机辅助软件工程（Computer Aided Software Engineering，CASE）成为研究热点，并开发了一些对软件技术发展具有深远影响的软件工程环境。

③ 20 世纪 80 年代中期～20 世纪 90 年代，出现了面向对象语言和方法，并逐渐成为主流的软件开发技术，开展了软件过程及软件过程改善的研究，注重软件复用和软件构件技术的研究与实践。

④ 20 世纪 90 年代以来，统一建模语言（Unified Modeling Language，UML）的提出和应

用，为开发团队提供了标准、通用的设计语言来开发和构建计算机应用。通过 UML，软件开发人员可以使用统一的语义和符号表示进行交流。

软件工程的发展史，反映了软件从简单到复杂，软件开发从个人行为到大型团队分工合作开发，软件开发工具和开发模式从粗糙到完善的发展历程。

在开发更多更强大软件的同时，对原有软件的纠错、优化和根据环境变化而调整也成为一项重要的工作，这些工作统称为维护。业界数据表明，60%～80%的工作耗费在软件首次交付给用户使用以后，因此在开发软件时就考虑到将来的软件维护可以有效地减少软件开发和维护的总时间。软件行业也一直致力于更加容易、快捷、低成本地开发和维护软件，如采用各种软件开发框架技术。

软件用于对信息的处理，如前所述，软件不仅仅意味着程序，也包含数据结构和文档。程序是开发语言指令的集合，通过执行这些指令可以满足预期的特征、功能和性能需求；数据结构指信息及信息的组织方式；文档指开发、维护和使用软件所必需的文字、图表等，它是软件工程的基础，为软件开发、维护和使用提供指导。

软件在功能越来越强大的同时，复杂性也急剧增加，从 DOS 1.0 的 4 000 行代码到 Vista 的 5 000 万行代码，软件开发早已从个人单打独斗过渡到团队作战。制定和实施开发计划，选用统一的开发规范与过程，协调用户、团队成员的沟通与合作，检查整个团队的开发进度，保证最终软件产品的质量等工作在软件开发中占有越来越重要的地位。

“软件工程”自产生以来，人们就寄希望于它去冲破“软件危机”这朵乌云。但是，软件危机现象并没有得到彻底排除，特别是一些老的危机问题可能解决了，但接着又出现了许多新的危机问题，于是不得不去寻找一些更新的工程方法。应该说，正是危机问题的不断出现，推动了软件工程方法学的快速发展。

## 1.2 软件危机的表现及产生的原因

### 1.2.1 软件危机的表现

某公司的一位主管收到了一张由计算机开出的零美元的账单，嘲笑了“愚蠢的计算机”后他将账单丢进了垃圾桶。一个月后，又一张标志着 30 天逾期的账单寄来了，这样的情形持续了 4 个月后，零美元账单带来了一封信，警告如果再不付账的话将会采取法律行动。由于担心自己的信用，这个主管在一个软件工程师的建议下，寄出了一张 0 美元的支票，最后一张 0 美元的收据送到了，该主管小心翼翼地将这张不同寻常的收据保存起来以备将来查询。

1991 年海湾战争中，一枚飞毛腿导弹穿过了爱国者反导弹的防御，击中了沙特阿拉伯 Dhahran 附近的一个兵营，造成 28 名美国人死亡，98 人受伤。这个错误是由累积的定时错误

引起的，爱国者导弹每次只能工作几小时，超过这个时间后，系统时钟就会复位。可悲的是新的软件第二天才运到。

20 世纪 90 年代，美国国内税收处让 Sperry 公司建立一套联邦税收表格自动处理系统，该系统被证明不适合当前的工作量，花费几乎是预算的两倍。到 1996 年，开发该系统共花费了 40 亿美元，但情况并没有得到改善，原因是“没有充分计划就错误行事”。

以上 3 个例子不过是失败的软件开发项目的冰山一角。总结起来，软件危机的主要表现如下。

（1）超出预算时间和成本

研究表明，每 8 个新的大型软件中就有两个会被取消，软件开发时间平均超出计划的 50%，而软件开发中的主要成本是人力资源成本，进度的落后意味着成本的增加。

（2）客户对生产出的软件不满意

开发人员往往不注重或不善于和客户交流，找出客户真正需要的东西，而是匆忙地进行开发，在开发过程中又不能从客户那里得到反馈信息，最后生产出的软件和客户想要的相差很远，难免出现纠纷。

（3）软件有残存的错误

研究表明，所有的大型软件系统中，大约 3/4 的软件系统有运行问题，不像预料的那样工作，或者根本不能使用。

（4）软件产品不可维护

由于设计时考虑不周和开发方法的原因，软件提交使用后不能改正错误，或在原有模块上不能增加新的功能，不能增加新的模块。

（5）文档资料不完整

软件文档是客户和开发人员之间的交流平台，也是软件开发团队的管理工具，必须和软件同步更新。但现实的情况是更改设计和修改代码时，有意无意地不对文档进行更新，造成文档和实际的代码不一致。

（6）软件生产率的提高跟不上硬件的发展速度

根据摩尔定律，每隔 18 个月计算机硬件的运算速度提高一倍，价格下降一半。但对于软件而言，现在的软件开发虽然引入了不少设计和开发的辅助工具，但总体而言仍以手工开发为主，与硬件的发展速度和对高质量软件的需求严重不符。

（7）软件成本在计算机系统总成本中的比例不断提高

软件开发成本变化规律如图 1.1 所示。

从图 1.1 可以看出，软件在整个计算机系统中所占的成分百分比从初期的不足 20%，到 20 世纪八九十年代超过 80%，呈逐年上升趋势；另一个需要注意的现象是，软件维护成本在软件开发总成本中的比例也越来越高。

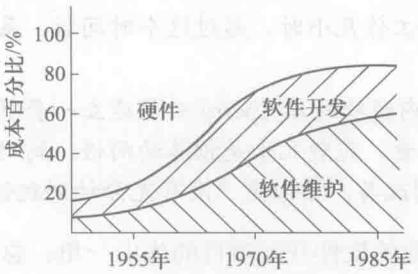


图 1.1 软件开发成本的变化

### 1.2.2 软件危机产生的原因

引起软件危机的主要原因如下。

（1）软件开发无计划性

开发者没有经过仔细考虑就匆忙动手，出现问题时才想办法补救，不能保证软件开发的进度和预算，不能保证软件质量，在进度落后时盲目增加人手，结果适得其反。《人月神话》一书生动地描述了这种情况："史前史中，没有别的场景比巨兽在焦油坑中垂死挣扎的场面更令人震撼。上帝见证着恐龙、猛犸象、剑齿虎在焦油中挣扎。它们挣扎得越是猛烈，焦油纠缠得越紧，没有任何猛兽足够强壮或具有足够的技巧能够挣脱束缚，它们最后都沉到了坑底"。

（2）软件需求不充分

开发者没有将需求弄清楚就匆忙上马，在开发过程中又不能和客户有效地沟通，许多问题在交付软件时才集中地爆发出来，这时已经是大势已去，难以挽回了。与数值计算软件和平时学习语言编写的程序不同，在实际软件开发中，首先应该满足客户的需要，而不是为了展示个人的技巧。

（3）软件开发过程无规范

开发过程没有统一的方法和规范，也不重视文档，各开发人员之间的接口没有统一规划，开发人员各自为战，导致最后软件不能融合成一个整体，在查找错误时也不能准确地定位。

（4）软件产品无评测手段

编程人员提交产品时没有进行测试，将软件测试看作是测试人员的事；模块之间的接口没有测试，模块间互相调用，出现错误时程序员互相推诿；整个软件系统没有进行整体测试，程序员只关心自己负责的部分；忽略压力及性能测试，整个软件系统效率低下或在大吞吐量动作时出现系统崩溃。

## 1.3 软件危机解决之道：软件工程

1968 年，北大西洋公约组织的计算机科学家们在联邦德国小城 Garmisch 召开的国际学术

会议上，第一次提出了“软件危机”这个名词，如图 1.2 所示。

图 1.2 1968 年 Garmisch 软件工程会议

1968 年秋季，北大西洋公约组织的科技委员会召集了近 50 名一流的编程人员、计算机科学家和工业界巨头，讨论和制定摆脱“软件危机”的对策。在此次会议上第一次提出了软件工程（Software Engineering，SE）概念，它是一门研究用工程化方法构建和维护有效、实用和高质量的软件的学科，它涉及程序设计语言、数据库、软件开发工具、系统平台、标准、设计模式等方面。

### 1.3.1 软件工程的定义

软件工程一直以来都缺乏一个统一的定义，很多学者、组织机构纷纷给出了自己的定义。

① 著名软件工程专家 Barry W. Boehm 定义软件工程：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。

② 美国电气电子工程师学会（Institute of Electrical and Electronics Engineers，IEEE）在 1993 年定义：软件工程就是将系统的、规范的、可度量的方法应用于软件的开发、运行和维护的过程，以及对该方法的研究。

③ 计算机科学家 Fritz Bauer 在 NATO 会议上给出的定义：建立并使用完善的工程化原则，以较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。

④《计算机科学技术百科全书》中的定义：软件工程是应用计算机科学、数学及管