软 件 工 程

〔美〕 R.S.普雷斯曼 著
郭肇德 郑少仁 译
杨芙清 校

国防工业出版社

内 容 简 介

本书讲述“软件工程”的技术，对整个软件工程进程的每一步骤作了简要的介绍。

全书共分十三章，前面六章介绍软件的计划阶段，第七章至第十一章转向软件的开发阶段，第十一章以后讨论软件测试技术、可靠性和软件的维护。

本书包括软件开发过程的管理和技术两方面的内容，可作为大学计算机专业高年级的学生或研究生的教材。对于从事软件工作的管理人员、系统分析员或程序员，本书是一本简明的指导性参考书。

SOFTWARE ENGINEERING—A PRACTITIONER'S APPROACH
Roger S. Pressman
McGraw-Hill Book Company 1982

*

软 件 工 程

〔美〕 R.S.普雷斯曼 著
郭肇德 郑少仁 译
杨芙清 校

*

国防工业出版社出版
（北京市车公庄西路老虎庙七号）
新华书店北京发行所发行　　各地新华书店经售
国防工业出版社印刷厂印刷

*

787×1092 1/16 印张17 1/2 402千字
1988年7月第一版　1988年7月第一次印刷　印数：0,001—6,000册

ISBN7-118-00050-7/TP4　　定价：5.60元

译者的话

自1968年在西德召开的NATO会议[1]上提出“软件工程”一词以来，短短十几年间，软件工程的发展十分迅速。当前，它已经成为计算机科学技术领域中的一个重要的研究方向，其研究领域涉及软件开发的各个方面。软件工程的主要目标是为克服伴随大型软件开发而出现的“软件危机”，提高软件的可靠性，为大型软件的开发研制寻求一条工程化、产品化的正确途径。

目前，我国软件开发规模日益扩大，“软件危机”的征兆已经显露，十分需要软件工程的理论指导。因此，我们把“Software Engineering A Practitioner's Approach”一书翻译出来介绍给读者。

本书内容广泛，深浅适宜。书中按照软件生命期的各个阶段，介绍了软件计划、软件设计、软件的实现和软件的测试与维护等各方面的内容。既从组织管理的角度，也从科学技术的角度介绍了软件工程的有关课题。本书是作为计算机软件专业高年级学生或研究生的教材而写的，但同时，也可以为从事软件工作的工程技术领导、系统分析员和程序员等在软件开发方面提供简明的指导。不失为软件工程方面的一本较新较好的书。本书各章后面均附有参考文献目录及大量的习题与思考题，可供教师和学生参考、选用。同时，在各章后面还附有进一步阅读的参考资料，以便读者进一步钻研感兴趣的有关课题。

由于软件工程方面的书籍在国内还不多见，许多问题的提法及专有名词的译名尚未统一，译文中不妥及失误之处在所难免，恳请读者批评指正。

译　者

一九八五年九月

---

[1] 1968年在西德加尔密斯（Garmisch）由北大西洋公约组织（NATO）主办的有关计算机软件的会议。——译者

目　录

序　言

从电子数字计算机发展的短暂历史看，五十年代和六十年代是硬件的年代。七十年代是一个过渡时期，或者说是软件被承认的年代。而现在我们已经处在软件的年代了。八十年代的处理机将具有巨大的能力，但事实上，计算技术的发展则很可能由于我们没有能力生产出能开发这些处理机巨大能力的高级软件而受到限制。

在过去的十年中，我们已经逐渐认识到通常被人们称之为“软件危机”的这样一个事实。软件的成本急剧地增长，成了许多以计算机为基础的系统的最大开支项目。这样的系统尽管其进度和完成日期可以事先确定，却很难如期完成。由于软件系统越来越大，其质量也变得越来越令人怀疑。负责软件开发项目的人员或机构很难控制项目的进程，他们能够参考借鉴的历史资料是有限的。

为了克服软件危机，近年来已经逐渐发展起一组总称为“软件工程”的技术。这些技术把软件作为一个工程产品来处理；它需要计划、分析、设计、实现、测试以及维护。这本教材的目的就是对整个软件工程进程的每一个步骤作一个简要的介绍。

本书的内容安排与软件生命期的各个阶段紧紧相呼应。前面的几章介绍计划阶段，重点放在系统的定义（计算机系统工程）、软件的计划和软件要求分析上。用于软件成本和进度估算的专门技术对计划管理人员、技术人员和学生们可能是特别重要的。

接下来的几章把重点转向软件的开发阶段。介绍软件设计的基本原理。此外，将详细介绍软件设计方法的两种重要类型，讨论各种软件工具，同时，提供对各种技术和工具的比较以帮助专业人员和学生使用它们，并从软件工程的角度着重介绍程序的编写风格。

后面的几章讨论软件测试技术、可靠性和软件的维护。论述与测试有关的软件工程各个步骤，介绍用于软件测试的专门技术。讨论软件可靠性预测的当前状况以及介绍关于可靠性模型和程序正确性研究方面的一般综述。最后一章将讨论管理问题和软件维护的技术状况。

这本书来源于 Bridgeport 大学高年级学生或一年级研究生软件工程课程教材。这门课程和这本教材包括了软件开发过程的管理和技术两方面内容。本教材的各章节与该课程的主要讲授课题大体相应。事实上，这本教材的一部分是由这些讲义加以整理而成的。因此，本书的写作体裁比较随便，所用的图也来自该课程中用到的一些幻灯图。

《软件工程》这本书可以以多种方式为各种读者所使用。对于实践管理人员、系统分析员或程序员，本书是软件工程方面的一本简明的指导性参考书，可作为大学高年级学生或研究生软件工程课程的基本教材，还可以给计算机科学系或计算机工程系大学生作为软件开发课程的补充读物。

在过去的十年中，软件工程方面的文献已经迅速地增长。我衷心地感谢许多为这门新学科的发展起过作用的作者。他们的工作对这本书和我的表述方法有着重要的影响。我也感谢帕特·杜兰（Pat Duran），利奥·兰伯特（Leo Lambert），基尤·李（Kyu

Lee)，约翰·穆萨（John Musa），克劳德·沃尔斯顿（Claude Walston），安东尼·沃塞曼（Anthony Wasserman），马文·泽尔科维兹（Marvin Zelkowitz）和尼古拉斯·兹维金佐夫（Nicholas Zvegintzov）（本书的审阅者们）以及彼德·弗里曼（Peter Freeman）（丛书编辑）。他们的富有创见性的见解和建议在本书的最后准备阶段是极其宝贵的。作者特别要感谢利奥·兰伯特和他的来自通用电气公司计算机管理部门的同事们，在我与他们长期的联系中他们向我介绍了他们广泛积累起来的经验。此外，作者也感谢 Bridgeport 大学的学生们和曾经短期参加过我所教的这门课程学习的数以百计的软件专业人员和他们的管理者们，他们和我的辩论、提出的问题和想法正是我们这本书所涉及的领域中的一些基本问题。

最后，我要感谢巴巴拉（Barbara），马休（Mathew）和米歇尔（Michael），是他们的支持才使本书得以出版。

R. S. 普雷斯曼

第一章　计算机系统工程

四百五十年前麦克维利[1]（Machiavelli）曾经说过：

> 没有什么事情比起率先引进某种事物的新秩序来得更困难、更冒风险、更没有把握获得成功……

在八十年代，以计算机为基础的系统将引进一种新的秩序。虽然科学技术已经有了巨大的进展，然而，麦克维利的话至今仍然是正确的。

软件工程——本书所讨论的这一课题——和硬件工程都可以看成是一门更广义的学科——计算机系统工程的一部分。上述各门学科都是为了在开发各种以计算机为基础的系统时引入某种规律性。

用于计算机硬件的工程技术是由电子设计技术发展起来的。而且，在三十多年的时间里已经达到了比较成熟的水平。硬件设计技术已经很好地确立，制造方法仍在不断地改进，可靠性则已是一种可以期待的现实而不再是一种愿望了。

不幸的是，计算机软件工程却还没有摆脱麦克维利所描述的困境。在以计算机为基础的系统中，软件已经取代了硬件而成为系统中设计起来最困难、最不容易成功（按时完成和不超过预计的成本）而且管理起来最危险的部分。随着以计算机为基础的系统在数量上、复杂程度上和应用范围上的不断增长，对软件的需求仍然有增无减。

用于计算机软件的工程技术只是在最近才得到广泛的承认。在五十年代和六十年代，人们把计算机程序设计看成是一种技艺。还不曾有过软件工程的先例，也没有采用过软件工程的方法。

时代正在变化！

1.1　计算机系统的发展

软件发展的历史是与计算机系统发展的三十年紧密相关的。更好的硬件性能、更小的体积以及更低的成本推动了更复杂的系统的形成。在三代的计算机发展阶段里，我们已经从电子管处理机发展到微电子学设备。在近来有关“计算机革命”的通俗读物中，奥斯本（Osborne）[1]认为八十年代的特征是“新的工业革命”而托弗勒（Toffler）[2]则把微电子学的出现统称之为人类历史上“第三次变革浪潮”的一部分。

图 1.1 画出了以计算机为基础的系统的发展过程，图中各个阶段按照应用的领域而不是按照硬件特征划分。在计算机系统发展的初期，硬件经历了不断的变化，而软件则被多数人作为一种事后的工作来看待。计算机程序设计被认为是一种“凭感触来驾驭”的技艺，对它而言，几乎没有什么系统的方法可以遵循。软件的开发实际上很难驾驭——往往进度推迟或成本一增再增。在此期间，多数系统采用批处理工作方式。只有一些交互式系统比如早期的美国飞机订票系统和国防方面的一些实时系统（如 SAGE 等）

---

[1] 麦克维利（1469～1527）——意大利政治家、资产阶级思想家。——译者

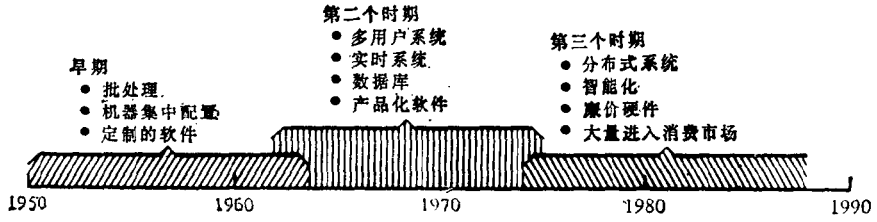

图1.1 计算机系统是如何发展的?

是例外。而在这些系统中硬件通常仅用来执行一个单一的程序，而这个程序又是为一个特定的应用目的而编制的。

在早期，通用硬件已经成为平常的事情了，而软件则仍然是为每一种用途分别设计，它们的通用性是有限的。

作为产品的软件（即为了销售给一个或多个用户而研制的程序）还处在它的幼年时期。大多数软件是由使用该软件的人或机构研制的。你写出它，使它运行，假如出了问题，也由你检查修改。由于作业很少变动，如果出了故障，只要你在现场，主管人员就可以放心去休息。这样，软件就带有个人的色彩，软件设计是在某个人的头脑中完成的一种隐含的过程。而且，往往没有软件说明书。

早些年，在如何实现以计算机为基础的系统方面我们学到了不少东西，而相对地在计算机系统工程方面还知道得很少。不过，公正地说，我们应当感谢在这个时期发展起来的许多杰出的以计算机为基础的系统给我们提供了经验。其中的某些系统现在仍在使用，而且作为这方面成就的里程碑至今仍受到人们的赞扬。

计算机系统发展的第二个时期（见图 1.1）跨越了从六十年代中期到七十年代中期这十年。多道程序设计、多用户系统引入了人-机对话的新概念。人机对话技术为计算机的应用开辟了新的天地并使硬件和软件提高到更为精湛的新水平。实时系统能够在几毫秒内而不是几分钟内收集、分析和变换来自多个信息源的数据，进而控制处理过程并产生输出。联机辅助存贮设备的进展导致第一次出现了数据库管理系统。

第二个时期也是以产品化软件的使用和“软件车间”的出现为特征的。在一个鼓励竞争的市场中，人们开发软件是为了广泛销售。来自产业界、政府部门和学术界的软件承包者们争相去“发展无所不包的软件包”并赚了大钱。

随着以计算机为基础的系统日益增多，计算机软件库开始膨胀。本单位自己的软件开发计划生产出几万条源程序语句。从外部购买来的软件产品又增加了几十万条新的语句。一片乌云出现在地平线上空，所有这些程序，即所有这些源语句，当检测出故障时就必须加以维护；当使用者的要求改变时，必须加以修改；或者要使软件在新购置的硬件上运行时，也必需要修改。软件维护的开销开始以令人恐慌的速度消耗着资财。更坏的是，许多软件所带有的个人色彩使得它们实际上是不可维护的。“软件危机”开始了。

计算机系统发展的第三个时期从七十年代初期开始，一直延续到八十年代初期。分布式系统（多个计算机、各机器并行执行和相互通信）极大地增加了以计算机为基础的系统的复杂性。由于微处理机和有关部件的功能越来越强而价格越来越低，因此，在最常用的计算机应用领域中，具有“嵌入智能”的产品取代了较大的计算机。

此外，微处理机的出现也使得人们可能以极低的成本实现复杂的逻辑功能。当前，那些熟悉硬件而对软件还比较生疏的技术人员正在采用这种技术作出新的产品。

硬件的迅速发展已经开始超过我们提供支持软件的能力。在第三个发展时期，软件危机日益严重。为了维护软件，消耗掉了数据处理预算的50%以上。而软件开发的生产率又跟不上新系统对软件需求的步伐。为了对付不断增长的软件危机，软件工程第一次得到认真的对待。

向计算机系统发展的第四个时期的过渡已经开始。具有1兆字节主存的16位和32位微处理机将为以计算机为基础的系统开辟全新的应用领域。从技术性应用向消费性市场过渡需要软件设计职业化，这只能通过计算机系统工程来完成。

1.2 计算机系统工程

计算机系统工程是一种问题求解活动。其主要内容是对系统所要求的功能加以揭示、分析，并把它们分配给系统的各个部分。计算机系统工程的概貌示于图1.2。系统分析和系统定义的技术将在第三章详细讨论。

在创建大多数新系统时，对系统所要求的功能往往只有模糊的概念。系统分析和系统定义的目的在于揭示摆在面前的项目的范围。这就要对需要进行处理的信息、所要求的功能、期望的性能以及设计的约束和检验的标准等进行系统的详细的分析。

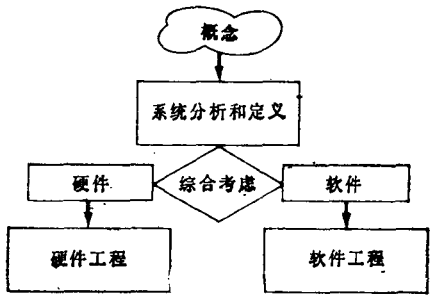

图1.2 计算机系统工程

在范围确定之后，计算机系统工程师必须考虑多种能潜在地满足项目范围的、可供选择的配置。下述几个方面的考虑可用来指导系统配置的选择：

1. 商业方面的考虑

该配置是否代表最可盈利的解法？它能成功地销售吗？最终的效益证明值得冒开发的风险吗？

2. 技术分析

有作出所有系统部件的技术吗？功能和性能能得到保证吗？该方案便于维护吗？技术资源存在吗？技术上要冒哪些风险？

3. 制造估计

生产设备和仪器是现成的吗？缺乏必要的部件吗？质量保证能实现吗？

4. 人员问题

有研制和生产方面的熟练人员吗？存在政治方面的问题吗？申请研制的人懂得该系统应具有的功能吗？

5. 与外部的接口

所提的配置与系统的外部环境有适当的接口吗？机器与机器、人与机器之间的通信方便易懂吗？

6. 法律方面的考虑

这个方案会导致非法的责任吗？专利权能得到适当保护吗？有没有被侵犯破坏的危险？

上述各个问题的重要程度将随各个系统而有所不同。

在综合考虑了各项因素之后，将选择一种配置，并将系统的功能分配给系统的各个组成部分。对一个以计算机为基础的系统而言，硬件、固件和软件是最可能被选择的一些组成部分。

1.3 硬件考虑

计算机系统工程总是将一项或多项系统功能分配给计算机硬件。下面几小节我们将讨论基本的硬件部件和应用。此外，还要介绍硬件工程的概况。

1.3.1 硬件的组成

计算机系统工程师选择硬件部件的某种组合构成以计算机为基础的系统的一个组成部分。硬件的选择并不是一件简单的事情，在选择时可以考虑以下几个因素：（1）一些部件应被组装成单独的构件块；（2）各部件之间的接口应当是标准的；（3）可以有许多不同的实现方案；（4）性能、成本和可用性相对说来应当比较容易确定。

图 1.3 给出了由“构件块”发展而来的硬件配置。用分立部件（即集成电路和晶体

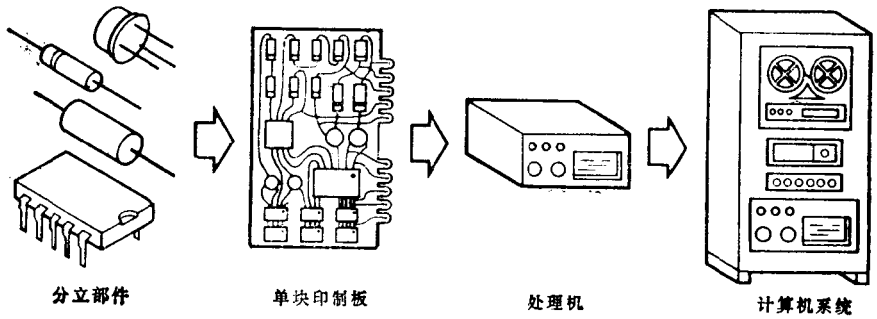

图1.3 硬件的配置

管、电容之类电子元件）装配成能实现一组特定操作的印制电路板。印制电路板被相互连接起来构成一个系统部件（例如：处理机和存储器），这些部件又被依次综合成硬件子系统或硬件系统部分。

对硬件配置的详细讨论超出了本书的范围。这些内容可以在许多参考文献〔例如文献 3～5〕中找到。对于那些不熟悉这方面内容的读者，我们将简要地探讨一下硬件配置中某些比较重要的组成部件。

硬件配置的基本组成部件可以在所有的计算机系统中找到。这些部件的系统结构（诸如各个部件的组织方式以及它们之间的通信路径等）有很大不同。图 1.4 示出了一种典型的硬件结构。中央处理机（CPU）与所有其它硬件部件相互配合实现各种算术的、逻辑的和控制的功能。CPU 的处理能力(以 MIPS——百万条指令/秒为单位度量)与系统的性能紧密相关。硬件的各个组成部分是用总线来相互连接的，总线作为通信路径传送各种指令、数据和控制信息。

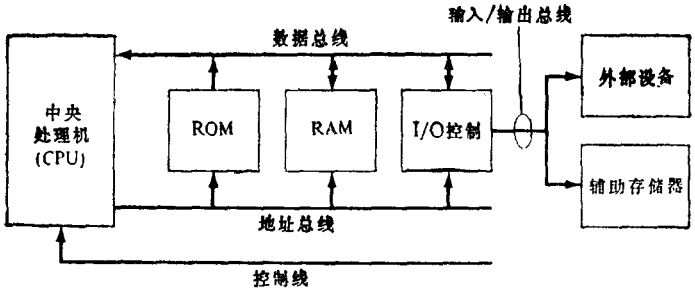

图1.4 典型的硬件结构

存储器为指令和数据提供存储的介质，它通过 CPU 执行的指令来直接地或间接地进行存取。主存储器可以定义为一种由 CPU 直接访问的存储介质。随机存取存储器 RAM(也称为读——写存储器）是一种最基本的存储器，对于所有要传送和存储数据的应用它是必不可少的。以微处理机为基础的系统可能只需要几百个字节的 RAM 存储器，而大型计算机则通常需要几百万个字节。只读存储器 ROM 则正像它的名字的含义那样只能被 CPU 读取。在制造只读存储器时，写入指令和数据，并把它永久地而且不可改变地固定下来。在电源去掉以后，其中的信息仍然保留。其它类型的 ROM(例如 PROM 和 EPROM）则可以用不太昂贵的微处理机开发系统来编入程序。只读存储器被广泛地用于各种消费产品、家庭用计算机和其它微处理机的应用领域。

辅助存储器是一种具有比主存储器容量更大和存取速度较慢的存储介质。最常用的辅助存储器称作磁盘，它以一个旋转的磁性介质为其特征。对磁盘而言，典型的信息存取时间是毫秒级，而容量则从 256000 字节到 600 兆字节以上。磁泡存储器是一种有希望极大地降低大容量数据存储器的成本和复杂性的固态辅助存储设备。磁带——一种最老式的辅助信息存储器——则继续被用作速度较慢但却便宜的档案数据存储介质。

对于一个以计算机为基础的系统来说，存储部件的选择关系到系统功能与性能的优劣。计算机系统工程师常犯的错误就是规定了容量太小的主存储器和（或)辅助存储器。因为存储器容量不够，常常造成功能降低、性能低劣和软件的价格昂贵的后果。由于存储器的成本正在迅速下降，因此，即使是为小批量生产的以计算机为基础的系统指定容量不足的存储器，也是不能原谅的。有关大批量生产的问题将在第三章讨论。

CPU 与外部世界的通信由输入-输出（I/O）或接口硬件处理。它的一些功能如下：接口硬件提供在 I/O 设备（外围设备）与 CPU 之间的专门的通信规程，控制信息的传输速率（它可能从每秒几十个字节到每秒几十万个字节），满足多种型号外设的接口标准（例如 RS-232C 和 IEEE-488）以及直接与其它的系统部件通信。

恰当的接口定义是计算机系统工程成败的关键。选择合适的接口硬件对于减轻系统合成的难度，简化 I/O 软件，提高处理机与外部世界通信的效率都有直接的影响。

1.3.2 硬件的应用

计算机硬件的应用可以分成三大类：信息处理；过程控制与实时应用以及嵌入智能。这些应用的发展过程大体上是与 1.1 节讨论过的计算机系统发展的三个时期相适应的。

今天，绝大多数以计算机为基础的系统把硬件用作为一个独立的信息处理机。信息被送入计算机系统进行分析和变换，可能还要得到一些其它的信息，然后产生结果。人们输入初始信息，而输出的结果信息亦被安排成适合于人们需要的格式。这方面的专门应用有商业数据处理、工程分析和数据管理等。

过程控制或实时应用是把硬件作为一个决策和控制的机构。硬件监测着过程的参数，并由软件进行分析，进而实施控制或作出报告。监测依靠机器或传感器的输入。硬件连续地监测一个过程（传感器输入）并对该过程不断地发出控制命令（反馈控制），在此同时从操作员处接受数据并向他们提供信息。典型的过程控制或实时应用常见于自动化生产系统（例如轧钢、炼油和化工处理）、系统和仪表控制、实时数据分析与报告等。应当说明一下，在过程控制或实时应用中，计算机硬件在位置上是可以与其它系统部件分开的。

把计算机硬件组装到一个更大的产品中时，这个系统就有了“嵌入智能”。几乎所有包含微处理机的产品都有嵌入的智能。其它的应用包括飞机上的飞行控制系统、各式各样的武器系统、灵巧的计算机终端和各种自动化应用等。这些计算机不象传统的计算机那样要放在空调房间的玻璃墙壁后面；反之，它们已与产品（系统）的其余部分完全组装在一起，一起飞，一起走，并经受同样的环境条件（例如温度、湿度和振动等）。

1.3.3 硬件工程

数字计算机的硬件工程是在几十年电子学设计经验的基础上发展起来的。硬件工程可以分为三个阶段：计划制定与规格说明阶段；设计与制造样机阶段以及生产、销售、安装与现场服务阶段。这些阶段示于图 1.5(*a*)、(*b*)、(*c*)。

一旦系统分析与系统定义完成之后，就把功能分配给了硬件。硬件工程的第一个阶段（图1.5（*a*））包括硬件研制计划与硬件要求分析。硬件研制计划旨在确定硬件研制工作的范围。也就是说，我们需要提出下列问题：

- 哪一类硬件可以最好地实现所确定的功能？
- 能采购到什么样的硬件？来源？能否买到？成本？
- 需要哪一种接口？
- 我们必须设计和制造些什么？可能存在什么问题？需要什么资源？

根据这些问题和其它一些条件，对硬件部分作初步的成本和进度估算。这些估算将由某些管理人员和技术人员进行审核，并作出必要的修改。

下一步，我们必须为硬件的设计和实现作出总的计划安排。硬件要求分析指为硬件部分的所有部件规定明确的功能、性能和接口要求。此外，还要确定某些限制条件（例如规模和环境）和测试标准。最后，要作出一份“硬件技术规格说明书”。在这一阶段，应该多次进行复审和修订。

工程的通俗形象是“甩开膀子干”。这表现在它的第二个阶段（图 1.5(*b*)）。在这一阶段，要分析所有的技术要求，并设计出一个初步的硬件方案。当设计向着详细的工程图纸（一种设计的规格说明）发展时，要进行技术复审。组件、部件都装配好，定做的部件也制成了，就可以把样机组装出来。对样机要作各种测试以确信它是否满足所有的要求。

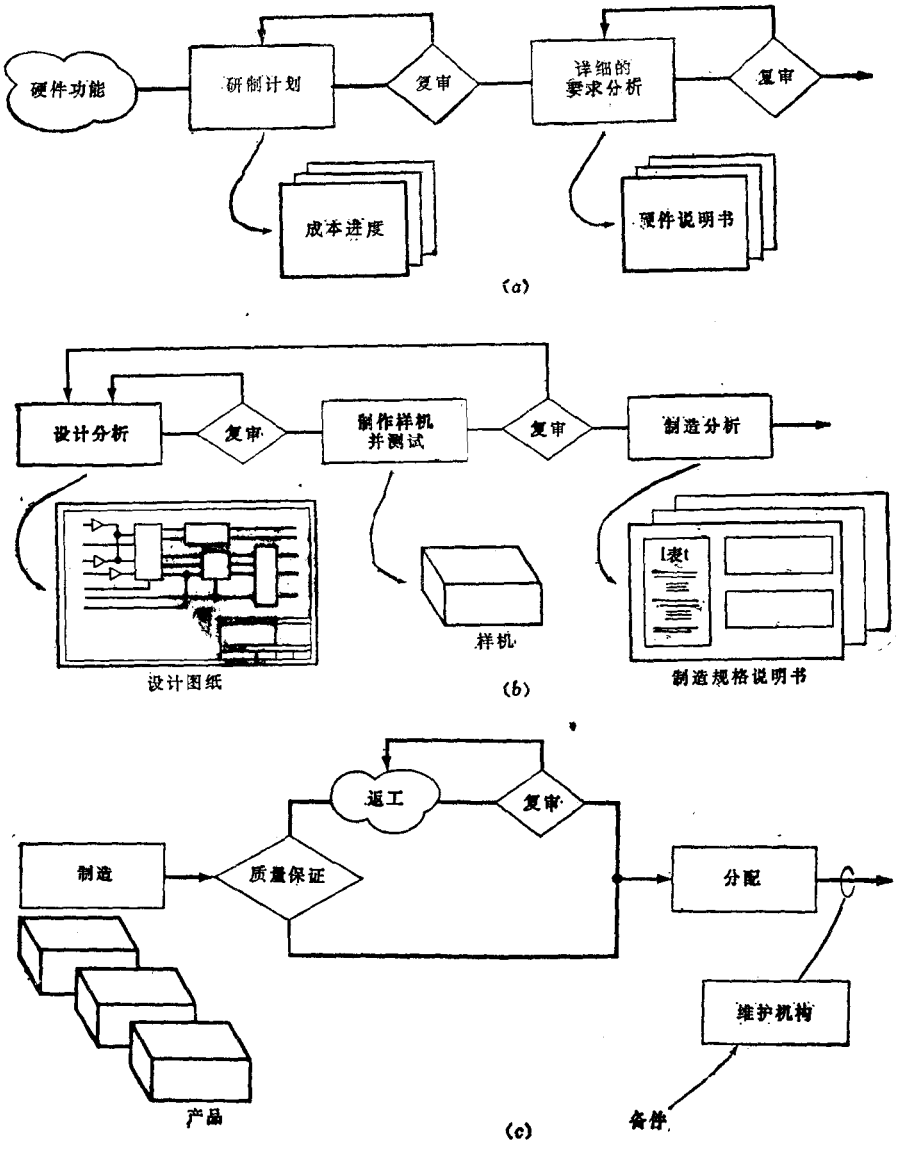

图1.5　硬件工程

（a）硬件工程-Ⅰ；（b）硬件工程-Ⅱ；（c）硬件工程-Ⅲ。

样机与所要制作的产品在外观上通常是不大相象的。因此，还需要编制出“制造规格说明书”。将试验电路板改为印制板，EPROM 或 PROM 改成 ROM，要设计新的机架，规定加工的工具和使用的设备。此时，工作的重点已从实现系统功能转向怎样易于制造。

硬件工程的第三个阶段对设计工程师几乎没有什么直接的要求，而对制造工程师的能力却提出严格的要求。在生产开始之前，要确定质量保证办法，并建立产品分配机构。为了维护和修理产品，要有库存备件，并建立现场服务组织。硬件工程的生产制造阶段示于图 1.5(c)。

1.4 软件考虑

计算机软件是一种逻辑的而不是物理的系统成分。因此，软件具有与硬件显著不同的特征：

- 对软件而言，没有明显的生产制造阶段；所有的成本都集中于计划与研制上。
- 软件不会磨损用坏；在软件的领域里几乎不存在备件！
- 软件的维护通常包括修改设计和提高功能。下面几小节，我们将讨论各种软件部件和各类语言，还将介绍软件工程的各个阶段。

1.4.1 软件的组成

计算机软件是信息，它有两种基本的形态：机器不可执行成分和机器可执行成分。我们在这一章中只介绍那些能直接产生机器可执行指令的软件成分。各软件成分如何构成一种配置将在后面几章讨论。

与机器可执行形式关系最密切的三种软件成分示于图 1.6。软件最终用语言的形式来规定软件的数据结构和过程属性。语言将由一个翻译程序来处理，它将该语言转换为机器可执行的指令。

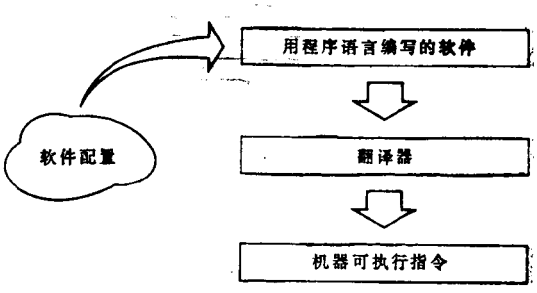

图1.6 软件成分

最理想的是，人类应当用自然语言（例如英语、西班牙语和俄语）与计算机对话。不幸的是，大量的词汇、复杂的文法以及要通过上下文才能理解的现象妨碍人类用自然语言与计算机通信。七十年代在语义信息处理和模式识别方面的研究[6]为使用自然语言作为人-机通信的媒介打下了基础。然而，至少在最近几年，用来说明程序的语言形式还仅限于“人工语言”。

所有的编程语言都是人工语言。每一种语言都有一组数量有限的词汇，有一种明确定义的文法以及构造良好的语法规则与语义规则。为了用机器进行翻译，这些属性都是必不可少的。软件成分的语言形式有机器级语言（也叫作汇编语言）和高级语言。

机器级语言是CPU指令集的符号表示。图 1.7 上摘录的微处理机汇编语言是一个例子。当一个优秀的软件设计人员编制一个便于维护、有良好文档资料的程序时，采用机器级语言可以非常有效地利用存储器和优化程序的执行速度。当一个程序被粗劣地设计而且文档资料很少时，则机器级语言将使出现的问题难以解决。

尽管机器级语言能提供很吸引人的执行速度并能节省存储空间，但它还是有许多严重的缺点：（1）程序的实现时间被拖延了；（2）编写的程序难读；（3）测试困难；（4）维护起来更加困难；（5）不同处理机之间的移植是不可能的。正象我们在第四章将看到的，当使用机器级语言时，软件的生产率将明显地下降。由于这些缺点都与机器级语言的使用有关，很可能在下一个十年中这种语言形式将被淘汰。

高级语言允许软件开发者以及设计出的程序与所用的机器无关。如果采用一个较复杂的翻译程序时，那么所用的高级语言的词汇、文法、语法和语义就可以比机器级语言