軟 件 工 程

—实践者的研究途径和方法

R. S. Pressman (美) 著

唐世渭 方 裕 译

徐家福 杨芙清 校

《小型微型计算机系统》编辑部

译 者 前 言

软件工程学是当前计算机科学的一个重要研究领域。为适应我国软件工程的迅速发展，我们翻译了Roger S.Pressman的

《SOFTWARE ENGINEERING

——A PRACTITIONER'S APPROACH》

（《软件工程——实践者的研究途径和方法》）一书。

但由于时间仓促，加上我们的水平有限，因此一定会有不少错误和不妥之处，敬请广大读者指正。

译 者

1984.6

前　言

在电子数字计算机的简短历史中，五十年代和六十年代是硬件的两个十年。七十年代是转折时期，是认识软件的时期。我们现在正面临着软件的十年。事实上，可能由于我们没有能力生产能够利用八十年代处理机巨大能力的高质量软件而使计算技术的进展受到限制。

在过去十年中，我们已经提高了对软件危机的认识。软件价格急剧上升，使之成为许多以计算机为基础的系统中花钱最多的项目。虽然制定了进度表和完成日期，但很少按期实现。当软件系统变得更大时，质量就变得更成问题了。负责软件开发项目的人员可用作指导的历史数据不多，并且对项目的进程难以控制。

总称为软件工程的一组技术已经针对软件危机而发展起来。这些技术把软件看作一种要求计划、分析、设计、实现、测试和维护的工程产品。本书的目的是对软件工程进程中的各步提供简要的介绍。

本书的目录完全平行于软件生存周期。前面几章介绍计划工作阶段，强调系统定义（计算机系统工程）、软件计划工作以及软件要求分析。关于软件价格及进度估算的具体技术，不仅项目管理人员而且从技术上进行实现的人员和学生都会特别感兴趣。

在后面的几章里，重点转移到软件开发阶段。介绍软件设计的基本原理。另外，详细介绍了两类重要的软件设计方法。讨论了各种软件工具，提供了方法之间和工具之间的比较，以便对实际工作者和学生们均有所帮助。在软件工程进程的前后联系中也强调了编码风格。

最后几章讨论软件测试技术、可靠性和软件维护。描述了与测试有关的软件工程步骤，并且介绍了软件测试的具体技术。讨论了关于软件可靠性预测的目前状况。并且概述了可靠性模型和程序验证方法。最后一章考察了软件维护的管理和技术两个方面。

本书是Bridgeport大学高年级学生或一年级研究生的软件工程课程的产物。课程和教科书包括了软件开发过程的管理和技术两个方面。书中各章大致对应于主要的讲授课题。事实上，本书部分是从这些讲课笔记的编辑版本引伸而来的。所以，书写风格是很随便的，并且图表也是从授课期间所用的视图得到的。

《软件工程：实践者的研究途径和方法》可以以多种方式为各种读者所使用。对于正在实际工作的管理员，分析员或程序员，本书可作为软件工程的简明指南。对于高年级本科生和研究生的软件工程课程，本书也可以作为基本教程。最后，对于在计算机科学或计算机工程专业本科生教学大纲中的早期软件开发，本书可用作补充指南。

在过去的十年中，软件工程的文献增长极为迅速。我非常感激那些对这门新学科发展有所贡献的许许多多的作者们。他们的工作对本书和我的陈述方式有重要的影响。我也希望感谢本书的审阅者：Pat Duran、Leo Lambert、kyu Lee、John Musa、Claude Walston、Anthony Wasserman、Marvin Zelkowitz、Nicholas Zvegintzov以及丛书编辑Peter Freeman。在最后的准备阶段，他们的深思熟虑的见解和建议是极为宝贵的。特别感谢Leo Lambert和他在通用电器公司计算机管理经营部的同事们，在我与他们的长期联系中，他们允许我利用了他们广泛的集体经验。另外，对于参加了我所教的短训班的

Bridgeport大学学生们和几百名软件专业人员以及他们的管理者们，我感谢他们的辩论、意见和询问，在一个象我们所讨论的这样领域中，这些都是必不可少的。

最后，我对Barbara、Mathew和Michael致以敬意和谢意，他们允许了第二卷的开始。

内 容 提 要

本书内容包括：C++程序设计基础；数据输入输出与程序基本结构；模块化程序设计；数组；结构体和共用体；指针等。

本书可作大专院校教材及自学者的参考用书。

责任编辑：朱桂兰

封面设计：赵一东

目　　录

第一章　计算机系统工程

四百五十年以前Machiavlli曾说过：

没有一种事情比带头引进关于事物的一个新规程来得更难处理，实施起来更危险或者更没有成功的把握……

在二十世纪八十年代，以计算机为基础的系统将引进一个新规程。从Machiavelli说上述话以来，虽然技术已经有了巨大进步，然而他的话听起来仍然很真切。

软件工程——本书所致力的课题——和硬件工程都是在我们称为计算机系统工程的更宽广的范畴中的活动。每一种规程表示企图以计算机为基础的系统开发中引进一种规程。

计算机硬件的工程技术由电子设计发展而来，并且在三十多年中已经达到了相当成熟的状态，硬件设计技术已经很好地建立起来，制造方法得到不断改进，可靠性已是一种现实的要求，而不是一种朴素的希望。

不幸的是，计算机软件仍然面临上面Machiavelli所描述的状况。在以计算机为基础的系统中，软件已经替代硬件而成为最难设计，最少可能成功（在时间上和成本方面）并且管理起来最危险的系统成分，随着以计算机为基础的系统在数量、复杂程度和应用方面的增长，对软件的需要将不断提高。

关于计算机软件的工程技术只是最近才得到普遍承认。本世纪五十年代和六十年代的时候，计算机程序设计还被看成是一种技艺，没有工程先例存在，也没有工程方法被应用。

时代正在变化！

1.1　计算机系统的发展

软件发展的历程是与计算机系统发展的三十年紧密相关的。较好的硬件性能、较小的体积和较低的成本已经促进了更精致复杂的系统的出现。在三代半的机器年代里，我们已经从电子管处理机进入了微电子设备。在最近流行的关于“计算机革命”的一些书中，Osborne〔1〕写道，80年代是一个“新的工业革命”的年代。Toffler〔2〕把微电子学的出现称作是人类历史中“第三次冲击波”的组成部分。

图1.1描述了以计算机为基础的系统按照应用领域而不是按照硬件特性的演变情况。在

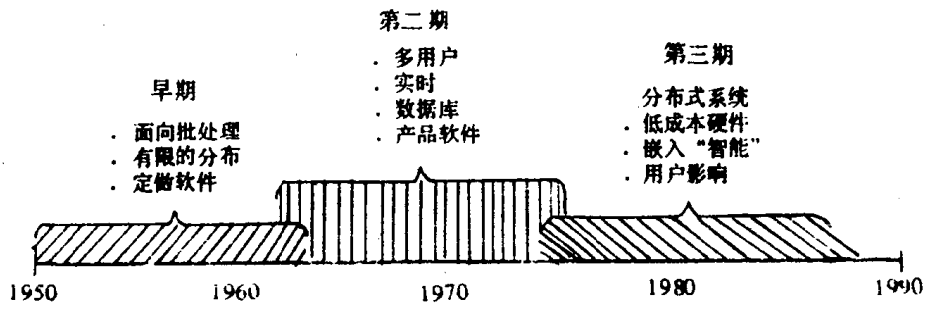

图　1.1　计算机系统是怎样发展的

计算机系统发展的早期，硬件经历了不断的变化而软件被看作是一个事后的事（“马后炮”）。计算机程序设计曾是一种“按裤子做座椅”的技艺，它很少有系统的方法存在。软件开发在进度拖延或者成本开始上升之前实际上是难以管理的。在这个时期大多数系统采用批处理。值得注意的例外是交互系统，例如早期的美国航空公司的预定系统和面向国防的实时系统（如SAGE）。然而，对大部分情况来说，是硬件致力于执行一个单个程序，而该程序又致力于一个特定的应用。

在早期，通用硬件是司空见惯的，而另一方面，软件却为每一个应用而专门设计，并且相对说来分布也很有限。

产品软件（即，为了卖给一个或多个顾客而开发的程序）当时正处于发展的初期。多数软件都是由相同人员或机构来开发和最终使用的。软件由你编写；由你使它运行；若是出了故障，也由你来修补。因为工作流动率很低，因而管理者可以保证：当遇到故障的时候，你会在现场。由于这种私人化的软件环境，设计是在人们头脑里执行的隐含过程．编制文件资料通常是不存在的。

早期我们学会了许多关于以计算机为基础的系统的实现方法，但相对来说很少学习计算机系统工程。然而，公正地说，我们必须承认许多杰出的以计算机为基础的系统都是在这个时期开发的。其中有些系统至今仍在使用并且提供了划时代的成就，这些成就继续证明这样的赞扬是当之无愧的。

计算机系统发展的第二期（图1.1）是横跨60年代中期到70年代中期的十年。多道程序设计，多用户系统引进了人机交互的新概念。交互技术开创了一个新的应用世界以及硬件和软件先进技术的新水平。实时系统能够从多个数据源收集、分析和传送数据，从而在几毫秒内而不是几分钟内控制过程并产生输出。联机辅助存贮设备的进步导致了第一代的数据库管理系统。

第二期也以产品软件的使用和“软件作坊”的出现为特征。为广泛分布于各种学科的需要而开发软件。一些人从工业、政府和研究部门出来改搞“开发最终软件包”而成为企业家并赚了一大笔钱。

随着以计算机为基础的系统的数量不断增长，计算机软件库开始扩大。在作坊中开发的项目生产了数万条源程序语句。从外边购买的软件产品加上了数十万条新语句。一块乌云出现在地平线上：当检测到故障时，由于用户要求变化而需要修改或采用购买的新硬件时，所有这些程序——所有这些源语句——都要进行维护。软件维护上所做的努力开始以惊人的速率吸收资源。还有更坏的情况，许多程序的私人化特点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期从70年代初期开始，持续到80年代初期。分布式系统——多台计算机、各机器平行执行和相互通信——极大地增加了以计算机为基础的系统的复杂性。随着微处理机和有关部件变得更强有力而且更便宜，带有“嵌入智能”的产品替代了较大的计算机而用于多数普通的计算机应用领域。

另外，微处理机的出现已经导致可以以很低的代价来使用复杂的逻辑功能。这个技术正在由技术人员汇集成产品，这些技术人员懂得硬件，但在软件领域里往往是初学者。

硬件的高速进步已经开始超越我们提供支持软件的能力。在第三期，软件危机加剧，软件维护吸收了数据处理预算的50%以上。并且软件开发的生产率无法跟上新系统所要求的速度。随着危机的增长，软件工程第一次被认真采纳。

向计算机系统发展的第四期的迁移已经开始，具有一兆字节主存的16位和32位微处理机将为以计算机为基础的系统打开至今尚未预见的应用领域。从一种技术到一种用户市场的跃迁要求专业人员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的活动。揭示、分析所要求的系统工程，并分配给各个系统元素。计算机系统工程过程的概述在图1.2中阐明。系统分析和定义的技术将在第三章中详细讨论。

多数新系统的创立在开始时对所要求功能还只能有一个模糊的概念。系统分析和定义的目的是揭示摆在面前的项目的作用范围。这个目标是通过对要处理的信息、要求的功能、希望的性能、设计约束和有效性准则进行系统地精化来得到的。

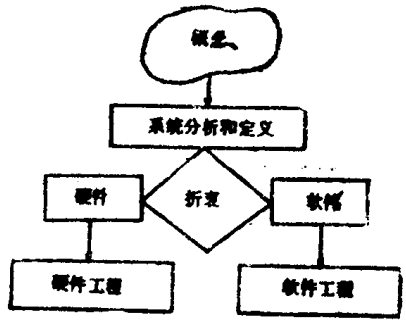

图 1.2 计算机系统工程

在作用范围已经建立起来以后，计算机系统工程师必须考虑一些潜在地满足作用范围的其它可能的配置，下面的折衷准则支配着系统配置的选择：

1.**商业考虑**。该配置是否代表多数有利可图的解法？能否成功的市场化？最终的获利将证明开发冒险是否值得？

2.**技术分析**。开发系统所有元素的技术是否存在？功能和性能是否有把握？该配置是否适于维护？技术资源是否存在？与技术有关的危险是什么？

3.**制造估价**。制造设施与仪器是否可用？必要的部件是否缺乏？质量保证能否充分实现？

4.**人的问题**。是否有为开发和制造而经过培训的可用人员？是否存在政治问题？要求者是否了解系统的功能？

5.**环境界面**。所建议的配置与系统的外部环境之间的接口是否合适？机——机和人——机之间的通讯是否按智能方式处理？

6.**法律考虑**，这个配置是否会导致违法的责任冒险？专利方面是否能够得到充分保护？是否存在着潜在侵犯？

上述准则的比重随系统而异。

在考虑了折衷以后，选择一个配置并且在可能的系统元素之间分配功能。对于一个以计算机为基础的系统，硬件、固件和软件是最可能被选择的元素。

1.3 硬件考虑

计算机系统工程总是分配一种或多种系统功能给计算机硬件。在以下的各段中将讨论基本的硬件部件和应用。另外将给出硬件工程的概述。

1.3.1 硬件部件

计算机系统工程师选择硬件部件的某种组合，这些硬件部件构成这个以计算机为基础的系统的一个元素。尽管选择硬件并不是一件简单的事情，但是可以借助下列特点来决定。（1）部件被组装成单独的构件块；（2）部件间的接口是标准的；（3）大量的“流行的”方案可以利用；以及（4）相对说来比较容易决定性能、成本和实用性。

图1.3给出了由“构件块”发展而来的硬件配置。离散的元件（即，集成电路和电阻、电容等电子元件）汇集成一块印刷电路板，它执行一组专门的操作。印刷电路板互相联接以构成系统部件（如，处理器和存储器），然后它们再接合成为硬件子系统或硬件系统单元。

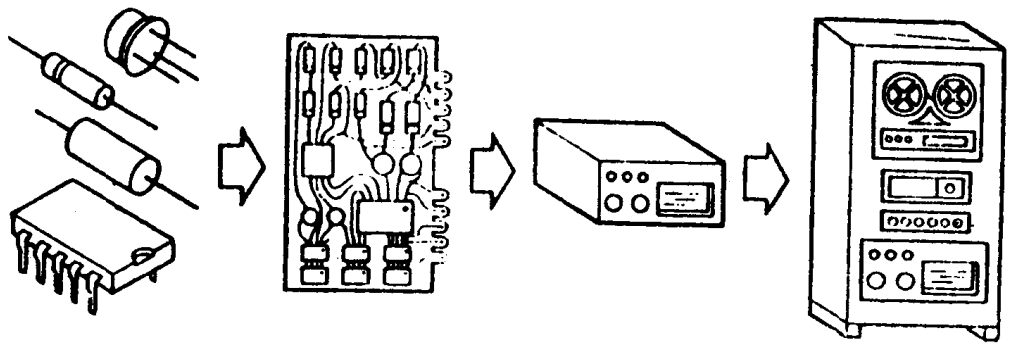

图 1.3 硬件配置

硬件配置的全面讨论超出了本书的范围，这方面的内容可以从许多参考文献〔如，3—5〕中找到。对于那些不熟悉这个科目的读者，我们将简要地说明硬件配置中某些较重要的方面。

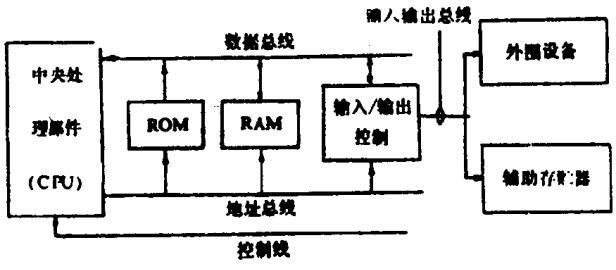

图 1.4 型道硬件系统结构

一个硬件配置的基本单元可以在所有计算机系统中找到。这些单元的系统结构变化极大。系统结构指的是这些单元的组织方式及其之间的通讯路径。图1.4阐明了一个典型的硬件结构。中央处理部件（CPU）实现算术逻辑和控制功能，它与所有其它硬件部件相互作用。CPU的处理能力，按每秒执行百万条指令来度量（MIPS），它与系统性能紧密相关。结构中各单元是通过总线传递指令、数据和控制信息的通讯路径来互相连接的。

存储器为指令和数据提供一个存储介质，并且由CPU通过执行指令来存取（直接的或间接的）。主存储器可定义为能由CPU直接寻址的一种存储介质。随机存取（也称读—写）存储器（RAM）对于所有要传送和存储数据的应用都是必要的。以微处理机为基础的系统可以只要求几百字节的RAM存储器，而大型计算机通常要求几兆字节。只读存储器（ROM），正如它的名字所指出的，只能被CPU读。只读存储器在制造时永久地（并且不可改变地）写上了指令或数据，并且在电源断电时保持这些信息。其它类型的ROM（如，PROM和EPROM）可以使用较便宜的微处理机开发系统为其编制程序。只读存储器已广

泛用于用户产品、家用计算机和其它微处理机应用领域中。

辅助存储器是一种比主存储器存取时间稍慢但有更大容量的存储介质。最普通的辅助存储器称作磁盘，它以一个旋转的磁介质为其特征。磁盘的信息存取时间一般在毫秒量级内，容量从256,000字节到600兆字节以上。磁泡存储器是一种固定状态的辅助存储设备，它可以大大减少大量数据存储的成本和复杂性。磁带，是最老形式的信息存储，仍在用作较慢但较便宜的档案存储介质。

对一个以计算机为基础的系统来说，存储器的选择决定功能和性能的成败。计算机系统工程师常常由于规定了太小的主存或辅助存储器而导致错误。存储器容量不足往往导致功能减弱、性能差和软件非常昂贵。因为存储器的成本正在迅速降低，所以对少数生产的以计算机为基础的系统的存储器容量不足是无法辩解的。与大容量产品相联系的问题将在第五章论及。

CPU与外界之间的通讯是通过输入—输出（I/O）或接口硬件来处置的，它的功能包括：接口硬件接纳I/O设备（外围设备）和CPU之间规定的通讯协议，控制信息传输率，传输率可以在每秒10个字符（字节）到每秒10万字符（字节）的范围内，满足多个卖主配置的接口标准（例如，RS—232C和IEEE—488），以及与其它系统部件进行直接通讯。

1.3.2 硬件应用

计算机硬件的应用可以分成三大类：信息处理、过程控制和实时应用；以及嵌入智能。这些应用的发展大致上与1.1节中所讨论的三个计算机系统时期相对应。

今天，绝大多数以计算机为基础的系统应用硬件作为一个孤立的信息处理机。信息输送到计算机系统进行分析或解释，然后可能获得其它信息，产生结果。原始的输入几乎总是来源于人，而且输出也按人的需要而格式化。专门的应用包括商用数据处理、工程分析和数据库管理。

过程控制/实时的应用把硬件集成为一种进行判定和控制的机构。硬件监控过程参数，并且采用通常由软件实现的试探法进行分析、控制和报告。监控的特点是机器或传感器的输入。硬件不断地监视一个过程（传感器输入），并且对过程产生控制命令（反馈控制），而同时从操作员处接收数据和向操作员提供信息。典型的过程控制/实时的应用通常表征为自动化生产（如，钢厂，石油提炼和化学过程），系统和仪器控制，实时数据分析和编制报表。应该注意，计算机硬件可以在地理上与其它部件相分离。

当计算机硬件组合在一个更大的产品中时，系统就有了“嵌入的智能”。几乎所有包含微处理机硬件的产品都有嵌入的智能。其它应用包括飞机航班控制系统，各种武器系统，“聪明”计算机终端和各种自动化应用。与比较通常的计算机硬件应用不同，计算机并不放在一个有空调房间的玻璃墙后面，而是它完全与产品（系统）的其余部分结合在一起，处于相同的环境（例如，热、湿和振动）中。

1.3.3 硬件工程

数字计算机的硬件工程是在数十年的电子设计的基础上发展起来的。硬件工程过程可以看作三个阶段：计划和规格说明；设计和样机实现；生产、分配和现场服务。图1.5a、b、c中阐明了这些阶段。

一旦进行了系统分析和定义，就把功能分配给硬件。硬件工程的第一阶段（图1.5a）包

和编制开发计划和硬件要求分析。编制开发计划以建立硬件作用范围。这就是说，我们要提出下列问题：

- 所指定的功能最好选择什么样的硬件？
- 可购买到什么硬件？来源、适用性和代价是什么？
- 要求何种接口？
- 我们必须设计和制造什么？潜在的问题和所要求的资源是什么？

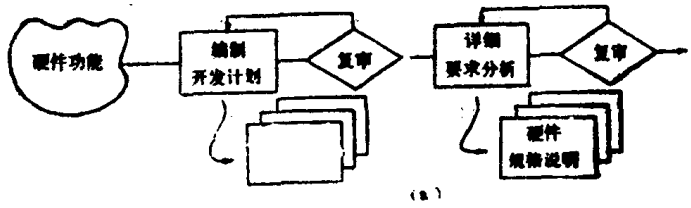

图 1.5a 硬件工程—Ⅰ

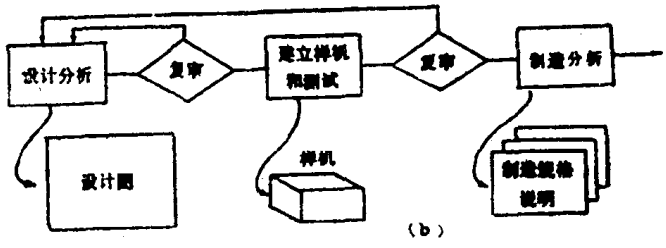

图 1.5b 硬件工程—Ⅱ

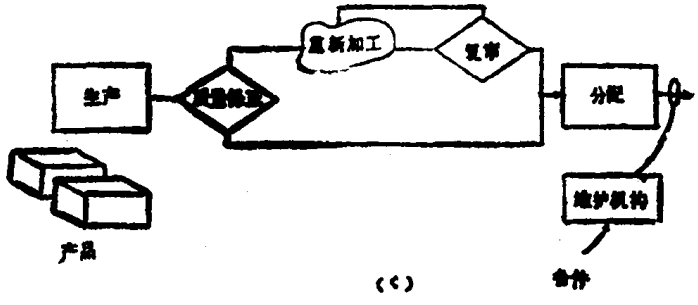

图 1.5c 硬件工程—Ⅲ

根据这些问题和其它一些问题，对硬件系统元素建立初步的成本和进度估计。这些估计由合适的管理者和技术人员进行复审，并且必要时进行修改。

其次，我们必须对硬件设计和实现建立一张“联络图”。进行硬件要求分析以便对所有硬件元素的成份规定明确的功能、性能和接口要求。另外，建立设计约束（如，规模和环境）和测试准则。通常产生一个硬件规格说明。在这阶段鼓励复审和修改。

“Shirt—Sleeve”工程的普遍图象可刻划为第二阶段（图1.5b）。分析要求，并且设计一个初步的硬件配置。当设计向着细节工程图（一个设计的规格说明）发展时，进行技术复审。获得成品部件，制造定做的部件，并且组装成样机。对样机进行测试以保证它符合所有的要求。

通常样机与工业产品很少相似，所以引伸出制造规格说明。试验电路板变成印刷电路板，EPROM或PROM变成ROM，设计新的插件，确定工具和仪器。重点从功能和性能移到易于制造。

硬件工程的第三阶段对设计工程师很少有直接要求而加重了制造工程师的负担。在生产开始之前，必须建立保证质量的方法，并且必须确定一个产品分配机制。把备用部件列入清单，为了产品的维护和修理建立一个现场服务机构。图1.5C阐明了硬件工程的制造阶段。

1.4 软 件 考 虑

计算机软件是一个逻辑的而不是物理的系统元素。所以，软件具有与硬件显著不同的特征：

- 对于软件没有明显重要的制造阶段；所有代价都集中在计划和开发上。
- 软件不会用旧；在软件世界中不存在什么备件！
- 软件维护通常包括设计的修改和提高改进。

在下面各段中，我们将考虑软件成份和语言种类。我们也介绍软件工程的各个阶段。

1.2.1 软件成份

计算机软件是以两种基本形式存在着的信息：机器不可执行的成份和机器可执行的成份。本章，为了我们讨论的目的，只介绍直接导致机器可执行指令的那些成份。若干软件成份构成一个配置，将在以后几章讨论。

与机器可执行形式最紧密的三种软件成份在图1.6中阐明。软件最终被转换成一种规定软件数据结构和过程属性的语言形式。该语言形式由一个翻译程序进行处理，把它转换成机器可执行的指令。

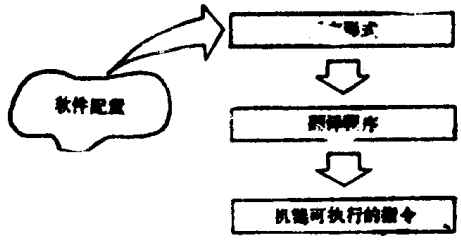

图 1.6 硬件成分

理想情况是人们能够用一种自然语言（例如，英语、西班牙语和俄语）与计算机通讯。不幸的是，大量的词汇、复杂的语法和我们为了了解而使用了上下文，这些都妨碍了通过自然语言来进行人——机通讯。二十世纪七十年代，对于语义信息处理和模式识别方面的研究为使用自然语言作为与计算机通讯的媒介初步奠定了基础。然而，至少在今后几年内，程序语言仍旧只限于人工语言。

所有程序设计语言都是人工语言。每种人工语言有一个不大的词汇表，一种显示定义的文法以及构造良好的语法和语义规则。对于机器翻译，这些属性都是基本的。软件成份的语言形式表征为机器级语言和高级语言。

由图1.7中给出的微处理机汇编语言摘录所阐明的机器级语言是CPU指令集的一种符号表示法。当一个好的软件开发者生产一个可维护的，有良好文档资料的程序时，机器级语言可以极端高效率地使用内存和优化程序执行速度。当一个程序设计得不好而且没有文档资料时，机器语言会使出现的问题趋于恶化。