我知道什么？

软件工程

〔法〕 雅克·普林茨 著
金维克 译
王 健 校订

科学出版社
www.sciencep.com

软件工程

〔法〕雅克·普林茨 著
全维克 译
王 健 校订

科学出版社
北京

图字：01-2003-0481 号

Que sais-je?

LE GÉNIE LOGICIEL

Jacques Printz

Presses Universitaires de France

6, avenue Reille, 75014 Paris

图书在版编目（CIP）数据

软件工程/〔法〕普林茨著；金维克译 .—北京：科学出版社，2005

（我知道什么?）

ISBN 7-03-014222-5

Ⅰ. 软… Ⅱ. ①普…②金… Ⅲ. 软件工程-普及读物 Ⅳ. TP311.5-49

中国版本图书馆 CIP 数据核字（2004）第 107228 号

责任编辑：沈红芬 姚庆爽 / 责任校对：张 琪

责任印制：钱玉芬 / 封面设计：张 放

科学出版社 出版

北京东黄城根北街16号

邮政编码:100717

http://www.sciencep.com

西源印刷厂 印刷

科学出版社发行 各地新华书店经销

*

2005 年 1 月第 一 版 开本：787×1092 1/32

2005 年 1 月第一次印刷 印张：5 1/8

印数：1—5 000 字数：89 000

定价：11.00 元

（如有印装质量问题，我社负责调换〈环伟〉）

前　言

软件工程是一门工程科学，其研究对象是信息化系统的生产。一般说来，信息系统是很复杂的，它的任务是在工业、管理、通信、国防等所有社会－经济系统中为各方面的信息处理提供必需的、优良的功能。

一个信息化系统里包括许多产地和性能各不相同的计算机，计算机之间用局域网（即企业内部的网络）和广域网（即企业之间的网络）相连，还有许多向周围收发信息的、功能各异的外围设备（如柜员机、雷达、机器人等）。

信息系统可以分为两大部分：

——硬件部分（计算机、终端设备、调制解调器、交换设备、传感器、执行机构等），它的作用是提供基本的处理能力和把信息系统与外界连接起来；

——软件部分，在各种信息处理和信息存储中，由它来保证必需的逻辑功能。

软件包括三种类型：

——设计软件，它对硬件有很强的依赖性；

——由软件出版商开发的软件包，它在一般情况

下是一些可以自行确定参数的黑盒子，可以提供这样或那样的具体功能；

——为满足企业某些特殊需求的软件，它可以由企业自行开发，也可以委托提供软件服务的公司来开发。

软件工程所关心的是各种不同软件的生产方法，以便保证以下几点：

——由开发商生产的软件要符合项目提出者（即代表最终用户的工程老板）的需求；

——保证软件的成本和开发期限在开始时确定的限度之内；

——在日后的软件发行中，要有效地遵守服务合同（包括软件性能、功能的可靠性、安全性等）。

正如“软件”的英文单词 software 所表示的，软件属于计算机系统中“软”的那部分，因为即使在软件系统的经营期间还可以根据新出现的需求而对它进行随意修改。这种可以不断进化的功能是软件非常重要的特性，因为这是影响系统生存周期（柔性、灵活性和易用性）以至折旧成本的决定性因素。

软件工程是确定软件生产过程各种规则和限制等必要条件的一门“元学科”。正像所有的元学科一样，它必须紧紧依靠软件开发的实践，始终对所提出的工具的有效性和合理性的条件进行研究。必须特别提防那些已经过时的模式和往往隐藏着思考上的漏洞的违背规律的东西，否则就将一事无成。

在这本薄薄的小册子里要介绍如此丰富的内容[7~9]，只能采取将问题尽量简化的方法，并且只能包括一些最核心的内容，而且我们将采用尽量直观的方式来进行讲述。

目　录

第一章
经济数据

一 软件成本

依照惯例，软件的生产成本是用“人月”或“人年”来计算的，但千万不要与软件开发所用时间的长短相混淆。3 名工程师用了 18 个月开发出的软件，其成本就是 54 人月或 4.5 人年。

一般说来，软件的“大小”或“容量”是用交付的或准备投入使用的软件所包含的源代码或指令的行数来表示的（单位为 ls，即行数；或 kls，即千行数）。这个数字指的是在计算机上可执行的部分，是软件中包含信息量多少的主要指标。

软件的开发能力同样用“行数”或“人月”来表示，说明软件的生产难度。

一些统计数据见下表。

软件性质	成　本	大　小	备　注
编译语言 - PASCAL，C - COBOL，FORTRAN - Ada	 10 人年 80～100 人年 150～200 人年	 20～30 千行 100～200 千行 ＞300 千行	开发时间 1～2 年 2～3 年 ＞3 年
关系数据库 (ORACLE, DB2, ……)	300～500 人年	300～600 千行	开发时间 3～5 年 其中包含测试第一版
大型实时控制系统 - 航天飞机 - SAFEGUARD① - SABRE②	 ＞1 000 人年 5 000 人年 955 人年	 2 200 千行 2 260 千行 960 千行	 用 HAL 写成，耗时 6 年 用 PL/1 语言写成，耗时 7 年 耗时 10 年，MTTF③ 为 55 小时
操作系统，设计系统 MVS④，VMS⑤，GCOS7……	2 500～5 000 人年	5 000～10 000 千行，20 世纪 70 年代以来，一般用高级语言编写	生存周期 15～20 年，其中第一版至少要用 5 年
工业系统 GPAO，MRP⑥，CIM⑦，……	300～500 人年	500～1 000 千行，重要的数据库	用 COBOL 或第四代语言编写
绘图软件（2D，3D）	150～300 人年	＞200 千行，一般用 FORTRAN 语言写成	其中包括一些伪代码算法
人工智能系统 - LISP，PROLOG - 专家系统	 10～20 人年 20～30 人年	一般用 C 语言，利用伪编译技术写成	这些语言一般用于需要推进优化的工业环境中

续表

软件性质	成　本	大　小	备　注
软件车间	100～200 人年	一般与词典连接	几乎包括开发周期的所有工具软件

注：①SAFEGUARD：美国从 20 世纪 70 年代开始研究的反弹道导弹防御系统。

②SABRE：由 IBM 公司研制的美国航空订票系统。

③MTTF：平均无故障时间。

④MVS：多重虚拟存储——译者注。

⑤VMS：虚拟网络服务——译者注。

⑥MRP：生产资源计划——译者注。

⑦CIM：计算机整合生产——译者注。

这些数字看起来似乎有些抽象，但如果将它们与日常生活中的事物联系起来，就能更清楚地了解它们所代表的极端复杂性。比如，一个 100 千行的软件，包括附件在内，其规模差不多相当于 10 本 400 页的图书。

这些庞大的数字仅仅是软件工程的第一批证据，它还不包括在软件维护和经营过程中继续发生的成本。所有生产效率的改进都将提高基本生产力带来的利益。如果考虑到软件的生存周期，那么还有两个新的困难：

——要赋予软件必需的、可以修改的特性；

——与人员变化有关的组织问题。

软件开发所对应的成本比预期的生存周期长短

（回收成本）更加重要。软件初始结构的质量对软件成本的影响极大。因此，笼统地讲，软件成本的计算办法是：成本＝开发＋维护＋经营，直到这个软件退出历史舞台。

二　成本结构

信息产业的成本结构已经发生了很大的变化。在开始阶段，硬件成本曾经占很大比重。成本结构的改变源自两次接踵而来的冲击：

——第一个冲击源自 IBM 公司的所谓“分别估价”政策，允许将软件作为成本进行分析；

——第二个冲击是在 20 世纪 80 年代微型计算机系统的兴起，它彻底改变了信息产业的面貌，大大降低了硬件的成本比重，这种现象在整个工业社会中还从未遇到过。

价格的下降，使得信息技术实实在在地大量深入到经济生活的各个领域。

这种变化来得极其迅猛，只用了 10 年左右的时间，就使得人们把软件生产摆在了信息产业的第一位。

三　需求的演变

在 20 世纪 80 年代，各种类型软件的需求都明显

地增加了。这种现象与各种因素有关，但所起的作用是一样的：

——计算机新应用的巨大需求；

——用户与信息系统之间日益加强的相互作用；

——以前所有模拟领域的数字化；

——不论在数量上还是容量上对信息服务需求的增长，尤其是在复杂系统的管理方面；

——诸如视频游戏和图像合成等新的应用领域的出现。

这涉及一种真正的“宏观现象”，表现为迫切需要在工艺上和方法上能使软件的生产越来越快，同时质量也要不断改进。

伴随着这种需求，软件在各种要求非常严格的任务系统中开始占优势。功能的可靠性因而成为软件工程中的头等问题。下面是若干风险方面的例子：

——属于人身安全的：民航的飞行控制、航空管理、高速列车等；

——属于经济安全的：交易所的信息系统、数字化电话交换中心、银行系统等；

——属于社会安全的：系统不能满足用户的需要和引起负作用，信息机密被系统所控制，非法使用者的新型犯罪形式（计算机病毒、“特洛伊木马”、逻辑炸弹等）都寻求使软件系统转向为他们谋利益。

软件功能的可靠性是一个严峻的挑战。因为与其他工程领域不同，软件的错误和缺陷既不会造成硬件

的失效，也不会出现其他技术领域中符合规律的“磨损”现象，其原因都是来自于编程活动中的人为的、固有的错误。

软件工程的主要目的之一，就是采取措施以保证在生产过程中清除掉一切错误，如果这一点做不到——尽管通常都是如此——也要保证软件中残留的差错（人们并不了解它们的数量）可以在系统任务执行时间允许的条件下通过专门的程序予以弥补。但在实时控制系统中，这段时间可能是极其短暂的（比如只有几毫秒）。

由于两种新出现的联合现象，这个目标变得越来越难于实现了：

——硬件功能的迅速提高，它的负作用是错误出现的频率增加了；

——计算机网络的逐步普及，这使得可以依靠程序模型而用非常简单的方式将应用联结起来，如分布和解惑客户机-服务器系统。这种连接增加了系统之间传染的机会，使得故障诊断变得非常困难，如果不增添软件过滤器，要诊断故障甚至是不可能的。

这种变化带来的影响是，检验和验证有效性成为软件成本中的主要部分。

第二章
软件的类型

一　困难的分类

在很长的一段时间里，人们曾经认为信息科学或信息技术是复杂的，而信息产业的管理却很简单。但随着网络和分布系统的大规模的出现，这种将两方面问题对立起来的看法在今天已经没有什么市场了。

人们还试图将软件按照大的应用领域来分类，试图反映其难以估量的复杂性。

领　域	数据结构	算　法	控　制
管理	困难	简单	简单①
数字分析，模拟	简单	困难	简单
电信	简单	简单	非常困难
实时控制	简单	困难	困难
数据处理与分析	非常困难	困难	简单
编译开发系统	困难	困难	困难

注：①客户机-服务器系统除外。

但这样的分类方法，除了在描述上还算有些可取之处外，设计师们对它却毫无兴趣，因为它并不能简化行动和决策。一种有效的分类方法必须反映某些深层的问题，显示出真实的困难，而且针对这种困难可以采取相应的对策：

——选择优秀的人才和优秀的组织；

——选择优秀的方法；

——选择优秀的工具。

于是，软件是否能够适应环境限制[1,2,17]的能力便显得非常清楚了。这种能力可以分为三个主要部分，就是上面提到的所有领域中的三类不同性质的程序。

——S 型程序是指软件特性完全确定和稳定的程序；

——P 型程序表示这样一类程序，它可能是不确定的，它的问题似乎与可以随着用户的需要而变化以实现最优化的标准有关；

——E 型程序要对环境刺激（包括人、传感器、其他系统等）做出反应，它可以是随机的，期限是任意的，并可能被差错所损害。

1.S 型程序

假设我们想编写一个计算数字 A 的平方根的程序。在各种著作中有许多方法可以做这种计算，如下

面就是牛顿提出的公式

$\sqrt{A} = \lim_{n\to\infty} X_{n+1} = 1/2(X_n + A/X_n)$　其中 $X_0 = A/2$。

这个公式就是程序的规范。它可能在任何范围都是有效的，并与环境没有任何牵连。相应的编程就是对这个公式的简单翻译，只需考虑计算机本身的限制(数字精度范围，一系列的收敛标准等)。有许多程序都属于这种类型，在现有的著作中也有各种不同的算法，可以解决这类非常广泛的问题[3]。这类程序的有效性仅取决于检验或证明是否准确地把公式翻译过来，但只有当公式不具有边缘效应的情况下才成立。

2.P 型程序

假设我们想编写一个可以将 Ada 语言写的文件翻译成机器语言的程序（这类程序叫编译语言)。尽管写入规则的数量是有限的，但这些规则是递归的，因此编译语言能够处理的文档数量是无限的。至少在理想状态下，编译语言翻译任何要处理的文件的速度应该是一致的。编译语言的使用方式可有很大不同：

——在程序调试阶段，受用户欢迎的是：最高的编译速度，丰富而精确的差错诊断功能，纠正差错时能避免每发现一个差错就要将程序重新编译一遍，在存储器中有程序的映像文件等。

——在最后要将程序交付生产的阶段，要去掉程序调试时的选项并尽可能以最优化的选项对程序进行