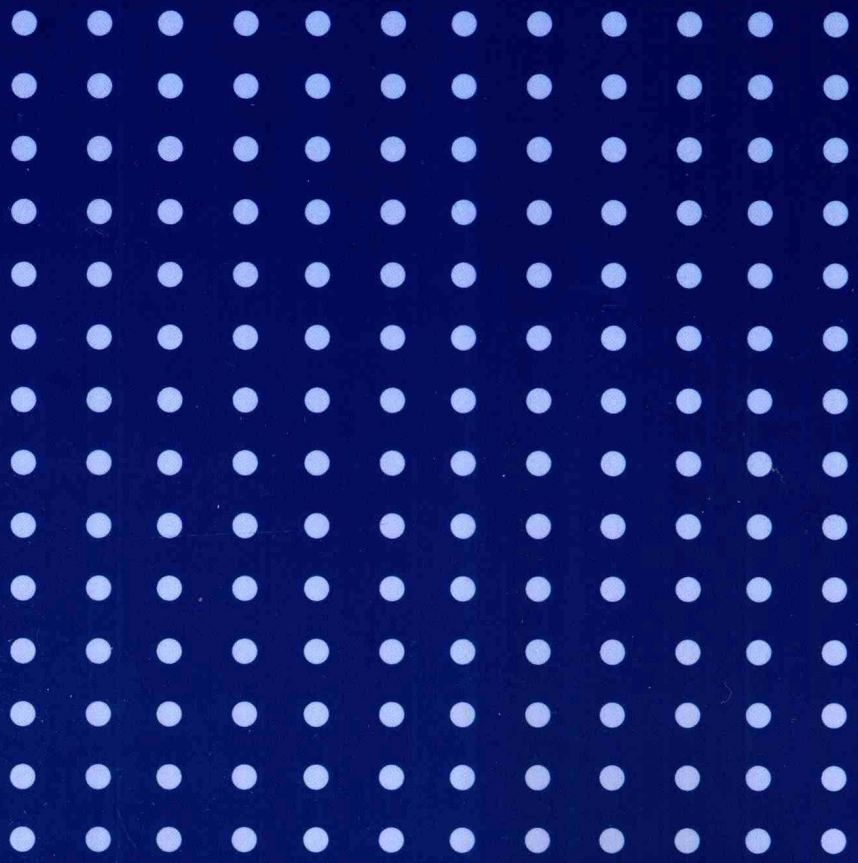
全国优秀畅销书
全国高校出版社优秀畅销书

张海藩 编著

软件工程

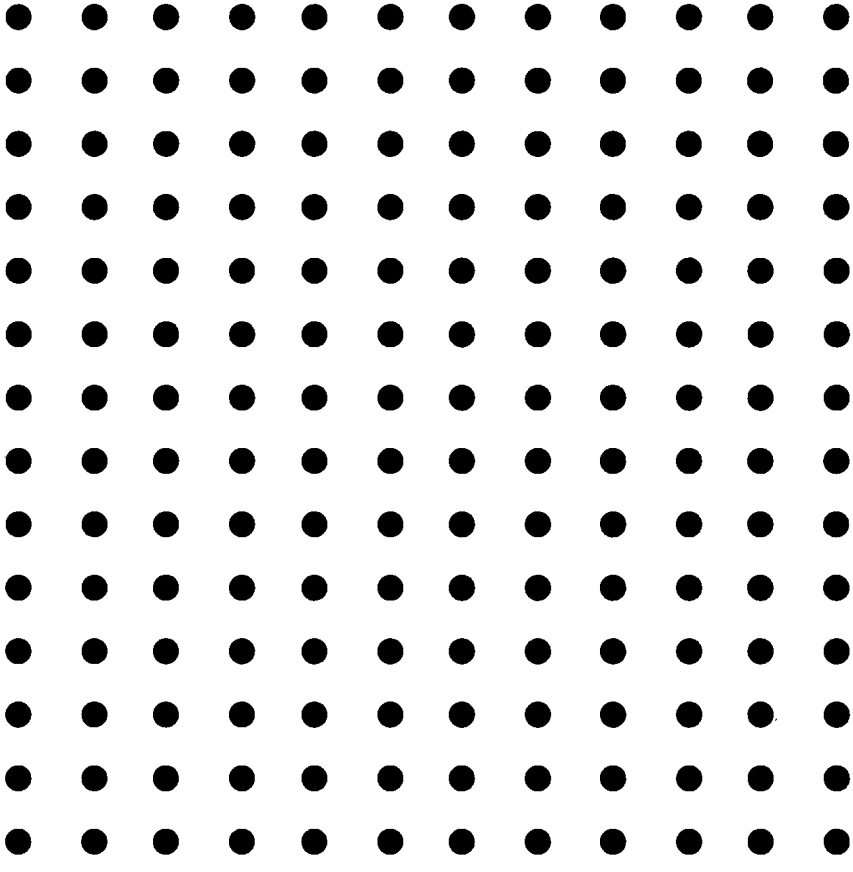


清华大学出版社

全国优秀畅销书
全国高校出版社优秀畅销书

软件工程

张海藩 编著



清华大学出版社
北京

内 容 简 介

为了满足“软件工程”课程学时较少的高等学校对一本适用教材的迫切需求，作者对先后荣获了全国普通高等学校工科电子类专业优秀教材一等奖和全国优秀畅销书奖，并被评为北京高等教育精品教材的《软件工程导论》作了精心改编，保留了原书中较重要、较新颖的内容，删除或简化了一些较陈旧或较次要的内容，写出了《软件工程》。

本书全面系统地讲述了软件工程的概念、原理和典型的方法学，并介绍了软件项目的管理技术。本书正文共13章，第1章是概述，第2章至第8章顺序讲述软件生命周期各阶段的任务、过程、结构化方法和工具，第9章至第12章分别讲述面向对象方法学引论、面向对象分析、面向对象设计和面向对象实现，第13章介绍软件项目管理。正文后面有两个附录，分别讲述了用面向对象方法和结构化方法开发两个软件的过程，对读者深入理解软件工程学很有帮助，也是上机实习的好材料。

本书可作为高等院校“软件工程”课程的教材或教学参考书，也可供有一定实际经验的软件工作人员和需要开发应用软件的广大计算机用户阅读参考。

图书在版编目（CIP）数据

软件工程 / 张海藩编著．—北京：清华大学出版社，2009.7
ISBN 978-7-302-19812-3

Ⅰ．软…　Ⅱ．张…　Ⅲ．软件工程—高等学校—教材　Ⅳ．TP311.5

中国版本图书馆CIP数据核字(2009)第045763号

责任编辑：袁勤勇
责任校对：时翠兰
责任印制：何　芊

出版发行：清华大学出版社　　**地　　址**：北京清华大学学研大厦A座
http://www.tup.com.cn　　**邮　　编**：100084
社　　总　　机：010-62770175　　**邮　　购**：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质　量　反　馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：北京鑫海金澳胶印有限公司
经　　销：全国新华书店
开　　本：185×260　　**印　　张**：22　　**字　　数**：506千字
版　　次：2009年7月第1版　　**印　　次**：2009年7月第1次印刷
印　　数：1～5000
定　　价：29.50元

前言

PREFACE

作者编著的《软件工程导论》已经出版了5个版本，累计销售了近一百万册，先后荣获全国普通高等学校工科电子类专业优秀教材二等奖和一等奖，并被评为全国优秀畅销书（前10名）和北京高等教育精品教材，国内许多高校把它选作“软件工程”课的教材。

但是，也有一些“软件工程”课程学时较少的高校教师反映，《软件工程导论》内容较多，用它作教材感到授课学时比较紧张。为了满足这部分学校的需求，作者对《软件工程导论》作了精心改编，保留了原书中较重要、较新颖的内容，删除或简化了一些较陈旧或较次要的内容，写出了《软件工程》。该书在保持原书结构不变的前提下把篇幅减少了近六分之一，将显著减少授课所需的学时。

编者

2009年5月

目录

CONTENTS

第1章 软件工程学概述

CHAPTER

迄今为止，计算机系统已经经历了4个不同的发展阶段，但是，人们仍然没有彻底摆脱“软件危机”的困扰，软件已经成为限制计算机系统发展的瓶颈。

为了更有效地开发与维护软件，软件工作者在20世纪60年代后期开始认真研究消除软件危机的途径，从而逐渐形成了一门新兴的工程学科——计算机软件工程学(通常简称为“软件工程”)。

1.1 软件危机

在计算机系统发展的早期时代(20世纪60年代中期以前)，通用硬件相当普遍，软件却是为每个具体应用而专门编写的。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(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的过程，除了程序清单之外，没有其他文档资料保存下来。

从20世纪60年代中期到70年代中期是计算机系统发展的第二个时期，这个时期的一个重要特征是出现了“软件作坊”，广泛使用产品软件。但是，“软件作坊”基本上仍然沿用早期形成的个体化软件开发方法。随着计算机应用的日益普及，软件数量急剧膨胀。在程序运行时发现的错误必须设法改正；用户有了新的需求时必须相应地修改程序；硬件或操作系统更新时，通常需要修改程序以适应新的环境。上述种种软件维护工作，以令人吃惊的比例耗费资源。更严重的是，许多程序的个体化特性使得它们最终成为不可维护的。“软件危机”就这样开始出现了！1968年北大西洋公约组织的计算机科学家在联邦德国召开国际会议，讨论软件危机问题，在这次会议上正式提出并使用了“软件工程”这个名词，一门新兴的工程学科就此诞生了。

1.1.1 软件危机的介绍

软件危机是指在计算机软件的开发和维护过程中所遇到的一系列严重问题。这些问题绝不仅仅是不能正常运行的软件才具有的，实际上，几

乎所有软件都不同程度地存在这些问题。

概括地说，软件危机包含下述两方面的问题：如何开发软件，以满足对软件日益增长的需求；如何维护数量不断膨胀的已有软件。软件危机具有长期性和症状不明显的特征。

具体地说，软件危机主要有以下一些典型表现：

- 对软件开发成本和进度的估计常常很不准确；
- 经常出现用户对“已完成的”软件产品不满意的情况；
- 软件产品的质量往往达不到要求；
- 软件通常是很难维护的；
- 软件产品往往没有适当的文档资料；
- 软件成本在计算机系统总成本中所占的比例逐年上升；
- 软件开发生产率提高的速度远远不能满足社会对软件产品的日益增长的需求。

以上列举的仅仅是软件危机的一些明显的表现，与软件开发和维护有关的问题远远不止这些。

1.1.2 产生软件危机的原因

在软件开发和维护的过程中存在这么多严重问题，一方面与软件本身的特点有关，另一方面也和软件开发与维护的方法不正确有关。

软件不同于硬件，它是计算机系统中的逻辑部件而不是物理部件。由于软件缺乏“可见性”，在写出程序代码并在计算机上试运行之前，软件开发过程的进展情况较难衡量，软件的质量也较难评价，因此，管理和控制软件开发过程相当困难。此外，软件在运行过程中不会因为使用时间过长而被“用坏”，如果运行中发现了错误，很可能是遇到了一个在开发时期引入的在测试阶段没能检测出来的错误。因此，软件维护通常意味着改正或修改原来的设计，这就在客观上使得软件较难维护。

软件不同于一般程序，它的一个显著特点是规模庞大，而且程序复杂性将随着程序规模的增加而呈指数上升。为了在预定时间内开发出规模庞大的软件，必须由许多人分工合作，然而，如何保证每个人完成的工作合在一起确实能构成一个高质量的大型软件系统，更是一个极端复杂困难的问题，这不仅涉及许多技术问题，诸如分析方法、设计方法、形式说明方法、版本控制等，更重要的是必须有严格而科学的管理。

软件本身独有的特点确实给开发和维护带来一些客观困难，但是人们在开发和使用计算机系统的长期实践中，也确实积累和总结出了许多成功的经验。如果坚持不懈地使用经过实践考验证明是正确的方法，许多困难是完全可以克服的，过去也确实有一些成功的范例。但是，目前相当多的软件专业人员对软件开发和维护还有不少糊涂观念，在实践过程中或多或少地采用了错误的方法和技术，这可能是使软件问题发展成软件危机的主要原因。

与软件开发和维护有关的许多错误认识和做法的形成，可以归因于在计算机系统发展的早期阶段软件开发的个体化特点。错误的认识和做法主要表现为忽视软件需求分析的重要性，认为软件开发就是写程序并设法使之运行，轻视软件维护等。

事实上，对用户要求没有完整准确的认识就匆忙着手编写程序是许多软件开发工程失败的主要原因之一。只有用户才真正了解他们自己的需要，但是许多用户在开始时并不能准确具体地叙述他们的需要，软件开发人员需要做大量深入细致的调查研究工作，反复多次地和用户交流信息，才能真正全面、准确、具体地了解用户的要求。对问题和目标的正确认识是解决任何问题的前提和出发点，软件开发同样也不例外。急于求成，仓促上阵，对用户要求没有正确认识就匆忙着手编写程序，这就如同不打好地基就盖高楼一样，最终必然垮台。事实上，越早开始写程序，完成它所需要用的时间往往越长。

一个软件从定义、开发、使用和维护，直到最终被废弃，要经历一个漫长的时期，这就如同一个人要经过胎儿、儿童、青年、中年和老年，直到最终死亡的漫长时期一样。通常把软件经历的这个漫长的时期称为生命周期。软件开发最初的工作应是问题定义，也就是确定要求解决的问题是什么；然后要进行可行性研究，决定该问题是否存在一个可行的解决办法；接下来应该进行需求分析，也就是深入具体地了解用户的要求，在所要开发的系统（不妨称之为目标系统）必须做什么这个问题上和用户取得完全一致的看法。经过上述软件定义时期的准备工作才能进入开发时期，而在开发时期，首先需要对软件进行设计（通常又分为概要设计和详细设计两个阶段），然后才能进入编写程序的阶段，程序编写完之后还必须经过大量的测试工作（需要的工作量通常占软件开发全部工作量的 40%～50%）才能最终交付使用。所以，编写程序只是软件开发过程中的一个阶段，而且在典型的软件开发工程中，编写程序所需的工作量只占软件开发全部工作量的 10%～20%。

另一方面还必须认识到程序只是完整的软件产品的一个组成部分，在上述软件生命周期的每个阶段都要得出最终产品的一个或几个组成部分（这些组成部分通常以文档资料的形式存在）。也就是说，一个软件产品必须由一个完整的配置组成，软件配置主要包括程序、文档和数据等成分。必须清除只重视程序而忽视软件配置其余成分的糊涂观念。

做好软件定义时期的工作，是降低软件成本提高软件质量的关键。如果软件开发人员在定义时期没有正确全面地理解用户需求，直到测试阶段或软件交付使用后才发现“已完成的”软件不完全符合用户的需要，这时再修改就为时已晚了。

严重的问题是，在软件开发的不同阶段进行修改需要付出的代价是很不相同的，在早期引入变动，涉及的面较少，因而代价也比较低；而在开发的中期，软件配置的许多成分已经完成，引入一个变动要对所有已完成的配置成分都做相应的修改，不仅工作量大，而且逻辑上也更复杂，因此付出的代价剧增；在软件“已经完成”时再引入变动，当然需要付出更高的代价。根据美国一些软件公司的统计资料，在后期引入一个变动比在早期引入相同变动所需付出的代价高 2～3 个数量级。图 1.1 定性地描绘了在不同时期引入一个变动需要付出的代价的变化趋势。

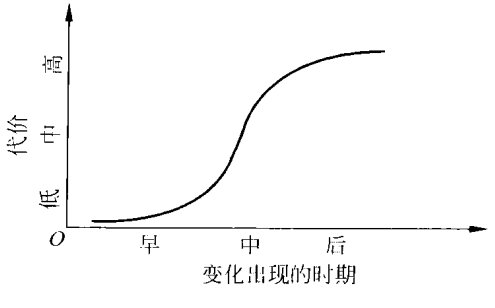


图 1.1　引入同一变动付出的代价随时间变化的趋势

通过上面的论述不难认识到，轻视维护是一个最大的错误。许多软件产品的使用寿命长达 10 年甚至 20 年，在这样漫长的时