软件工程

教学做一体化教程

陈恒 王雅轩 景雨◎编著

清华大学出版社

软件工程

教学做一体化教程

陈恒 王雅轩 景雨◎编著

清华大学出版社
北京

内 容 简 介

本书采用“教学做”一体化的方式撰写，并将每章内容分解为核心知识、能力目标、任务驱动、实践环节四个模块。全书共10章，第1章是软件工程基本概念，第2～7章顺序介绍了软件生命周期各阶段的任务、过程、结构化方法和工具，第8章讲述了面向对象方法学，第9章介绍了软件项目管理，第10章给出了经典的软件工程实验。书中实例侧重实用性和启发性，通俗易懂，使读者能快速掌握软件工程的基础知识与项目管理技能，为适应实战应用打下坚实的基础。

本书适合作为高等院校“软件工程”课程的教材或教学参考书，也适合作为有一定经验的软件工程人员的参考用书。

图书在版编目(CIP)数据

软件工程教学做一体化教程/陈恒，王雅轩，景雨编著. —北京：清华大学出版社，2013
ISBN 978-7-302-32422-5

Ⅰ. ①软…　Ⅱ. ①陈… ②王… ③景…　Ⅲ. ①软件工程－教材　Ⅳ. ①TP311.5

中国版本图书馆CIP数据核字(2013)第105239号

责任编辑：田在儒
封面设计：李　丹
责任校对：李　梅
责任印制：王静怡

出版发行：清华大学出版社
网　　址：http://www.tup.com.cn，http://www.wqbook.com
地　　址：北京清华大学学研大厦A座　　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
课件下载：http://www.tup.com.cn,010-62795764

印 装 者：北京国马印刷厂
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张**：15　　**字　　数**：362千字
版　　次：2013年8月第1版　　**印　　次**：2013年8月第1次印刷
印　　数：1～3000
定　　价：36.00元

产品编号：045669-01

前言
FOREWORD

本书按照“教学做”一体化模式精编了软件工程的核心内容，以核心知识、能力目标、任务驱动和实践环节为模块组织本书的体系结构。核心知识体现最重要和实用的知识，是教师需要重点讲解的内容；能力目标提出学习核心知识后应具备的能力；任务驱动给出了教师和学生共同来完成的任务；实践环节给出了需要学生独立完成的实践活动。

全书共10章。第1章概括地介绍了软件工程基本概念，包括软件、软件危机、软件工程、软件生命周期与常用模型。第2～7章按软件生命周期的顺序讲解了各阶段的任务、过程、方法和工具，其中：第2章重点讲述了如何使用系统流程图和数据流图分别描绘系统的物理模型和逻辑模型；第3章是需求分析与建模的有关知识，包括需求分析过程、需求获取方法、结构化分析建模工具以及软件需求规格说明书的内容框架；第4和5章是有关软件设计的知识，详细地介绍了软件设计的原理、工具、方法和文档，包括模块化设计原理、软件结构及描绘它的图形工具、面向数据流的设计方法、面向数据结构的设计方法以及设计文档的内容框架；第6章是关于系统实现的知识，重点讲述了系统实现的原理、技术和方法，包括编码、单元测试、集成测试、白盒测试、黑盒测试以及JUnit单元测试工具；第7章是有关软件维护的知识，包括维护策略与方法；第8章系统地讲解了面向对象方法学的有关知识，包括面向对象的基本概念、面向对象分析建模的原理与方法、面向对象程序的设计模式以及设计模式的应用；第9章讲述了软件项目管理的概念、原理、方法与技术，包括成本管理、进度管理、配置管理、风险管理、过程管理以及管理工具Microsoft Project的应用；第10章由5个实验组成，其目的是训练综合运用知识的能力，巩固本书前9章所学知识，提高工程实践与管理的能力。

本书特别注重引导学生参与课堂教学活动，适合高等院校相关专业作为教、学、做一体化的教材，也可以供软件工程爱好者、从业者自学使用。

编　者

2013年4月

目 录
CONTENTS

第1章

软件工程基本概念

主要内容

- 软件危机与软件工程
- 软件生命周期
- 软件工程的常用模型

随着计算机科学技术的迅速发展，如何更有效地开发软件产品越来越受到人们的重视。同时，由于软件复杂程度的不断增加，开发和维护的一系列严重问题（软件危机）随之产生。软件工程正是致力于解决软件危机，研究如何更有效地开发和维护计算机软件的一门新兴学科。

本章将介绍软件工程的基本概念，包括软件、软件危机、软件工程、软件生命周期与常用模型等。

1.1 软件危机与软件工程

1.1.1 核心知识

1. 计算机软件的发展

随着计算机硬件性能的极大提高和计算机体系结构的不断更新，计算机软件系统更加成熟和更为复杂，从而促使计算机软件的角色发生了巨大的变化，其发展历史大致可以分为如下四个阶段。

第一阶段是20世纪50年代初期至20世纪60年代初期的十余年，是计算机系统开发的初级阶段。计算机软件实际上就是规模较小的程序，程序的编写者和使用者往往是同一个（或同一组）人。由于程序规模小，程序编写起来比较容易，也没有什么系统化的方法，对软件的开发过程更没有进行任何管理。这种个体化的软件开发环境使得软件设计往往只是在人们头脑中隐含进行的一个模糊过程，除了程序清单之外，没有其他文档资料。

第二阶段跨越了从20世纪60年代中期至20世纪70年代末期的十余年，多用户系统引入了人机交互的新概念，实时系统能够从多个源收集、分析和转换数据，从而使得进程的控制和输出的产生以毫秒而不是分钟来运行，在线存储的发展产生了第一代数据库管理系统。

第三阶段是20世纪70年代中期至20世纪80年代末期，分布式系统极大地提高了计算机系统的复杂性，网络的发展对软件开发提出了更高的要求，特别是微处理器的出现和广泛应用，孕育了一系列智能产品。硬件的发展速度已经超过了人们对软件的需求速度，使得硬件价格下降，软件的价格急剧上升，导致了软件危机的加剧，致使更多的科学家着手研究软件工程学的科学理论、方法和时限等一系列问题。软件开发技术的度量问题受到重视，最著名的有软件工作量估计COCOMO模型、软件过程改进模型CMM等。

第四阶段是从20世纪80年代末期开始的。这个阶段软件体系结构从集中式的主机模式转变为分布式的客户机/服务器模式(C/S)或浏览器/服务器模式(B/S)，专家系统和人工智能软件从实验室走出来进入了实际应用，完善的系统软件、丰富的系统开发工具和商品化的应用程序的大量出现，以及通信技术和计算机网络的飞速发展，使得计算机进入了一个大发展的阶段。

2. 计算机软件的定义及特点

软件是计算机系统中与硬件相互依存的一部分，包括程序、数据及其说明文档。其中程序是能够完成特定功能的指令序列；数据是程序能正常操纵信息的数据结构；文档是与程序设计、开发及维护有关的各种图文资料。

软件同传统的工业产品相比，具有以下特点。

(1) 软件是一种逻辑产品。软件产品是看不见摸不着的，因而具有无形性，是脑力劳动的结晶，是以程序和文档的形式出现的，保存在计算机存储器和光盘介质上，通过计算机的执行才能体现其功能和作用。

(2) 软件产品的生产主要是研发，软件产品的成本主要体现在软件研发所需要的人力上。软件一旦研发成功，通过复制就产生了大量软件产品。

(3) 软件在使用过程中，没有磨损、老化的问题。但在使用过程中为了适应硬件环境以及需求的变化需要进行修改。当修改的成本变得难以接受时，软件即被抛弃。

(4) 软件的开发主要是脑力劳动。

(5) 软件会越来越复杂。软件涉及人类社会的各行各业、方方面面，软件开发常常涉及其他领域的专门知识，这对软件工程师提出了很高的要求。

(6) 软件的成本相当昂贵。软件研发需要投入大量、高强度的脑力劳动，成本非常高，风险也很大。

(7) 软件工作牵涉很多社会因素。软件的开发和运行涉及机构、体制和管理方式等问题，还会涉及人们的观念和心理等因素。

3. 软件危机与软件工程

20世纪60年代中期，大容量、高速度计算机的出现，使计算机的应用范围迅速扩大，软件开发急剧增长。软件系统的规模越来越大，复杂程度越来越高，软件可靠性问题也越来越突出。原来的个人设计、个人使用的方式不再能满足要求，迫切需要改变软件生产方式，提高软件生产率，软件危机爆发。事实上，软件危机几乎从计算机诞生的那一天起就出现了，只不过到了1968年，北大西洋公约组织的计算机科学家在联邦德国召开的国际学术会议上第一次提出了“软件危机”这个名词。

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。这类问题

绝不仅仅是“不能正常运行的软件”才具有的，几乎所有软件都不同程度地存在这类问题。概括来说，软件危机包含两方面问题：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产品。

具体地说，软件危机主要有下列典型表现。

(1) 软件开发进度难以预测，软件开发成本难以控制。开发成本超出预算，实际进度比预订计划一再拖延。

(2) 用户对产品功能难以满足。

(3) 软件产品质量无法保证。

(4) 软件产品难以维护。

(5) 软件缺少适当的文档资料。

(6) 软件的成本不断提高。

(7) 软件开发生产率的提高赶不上硬件的发展和人们需求的增长。

之所以出现软件危机，其主要原因一方面是与软件本身的特点有关；另一方面是与软件开发和维护的方法不正确有关。

为了消除软件危机，既要有技术措施，又要有组织管理措施。软件工程正是从技术和管理两方面研究如何更有效地开发和维护计算机软件的一门新兴学科。

1968 年秋季，第一届 NATO(北约)会议上第一次提出了软件工程这个概念。概括地说，软件工程是一门指导计算机软件开发和维护的学科。它采用工程的概念、原理、技术和方法来开发与维护软件，把先进的、正确的管理理念和当前最好的技术结合起来，以最小的经济代价开发出高质量的软件并维护它。

1.1.2 能力目标

了解计算机软件的发展历史，理解软件危机产生的原因，掌握软件、软件危机以及软件工程的概念。

1.1.3 任务驱动

任务 1：试想自己平时写的计算机应用程序，是计算机软件吗？

任务 2：在平时生活中，你使用过哪些软件？它们出现过问题吗？请举例说明。

1.1.4 实践环节

请上网查阅资料，了解软件工程与传统工程的区别是什么。

1.2 软件生命周期

1.2.1 核心知识

同任何事物一样，软件也有其孕育、诞生、成长、成熟和衰亡的生存过程，一般称为“软件生命周期”。软件工程采用的生命周期方法学就是从时间角度对软件开发和维护的复杂问题进行分解，把软件生命周期划分为软件定义(软件计划)、软件开发和软件维护 3 个时期，

每个时期又划分为若干个阶段。每个阶段的任务相对独立，而且比较简单，便于不同人员分工协作，从而降低了整个软件开发工程的困难程度；在软件生存周期的每个阶段都采用科学的管理技术和良好的技术方法，而且在每个阶段结束之前都从技术和管理两个角度进行严格的审查，合格之后才开始下一阶段的工作，这就使软件开发工程的全过程以一种有条不紊的方式进行，保证了软件的质量，特别是提高了软件的可维护性。

1. 定义时期

定义时期主要是确定待开发的软件系统要做什么；确定系统开发是否成功；弄清系统的关键需求；估算软件开发的成本；制定软件开发进度表。这个时期的工作通常又称为系统分析，由系统分析员负责完成。定义时期通常进一步划分成三个阶段，即问题定义、可行性研究和需求分析。

（1）问题定义

系统分析员通过对实际用户的调查，提出关于软件系统的性质、工程目标和规模的书面报告，同用户协商，达成共识。

（2）可行性研究

系统分析员需要制订软件项目计划，包括确定工作域、风险分析、资源规定、成本核算、工作任务和进度安排等。

（3）需求分析

对待开发的软件提出的需求进行分析并给出详细的定义。开发人员与用户共同讨论决定哪些需求是可以满足的，并对其加以确切的描述。这个阶段的一项重要任务是用正式文档准确地记录系统的需求，这份文档通常称为需求规格说明书。

2. 开发时期

开发时期主要是确定待开发的软件应怎样设计与实现，这个时期通常由概要设计、详细设计、编码和单元测试以及综合测试组成。总体设计与详细设计又称为系统设计，编码和单元测试与综合测试又称为系统实现。

（1）概要设计

概要设计又称为总体设计。这个阶段的主要任务是设计程序的体系结构，即确定程序由哪些模块组成以及模块间的关系。

（2）详细设计

详细设计又称为过程设计或模块设计。这个阶段的主要任务是设计出程序的详细规格说明，即确定实现模块功能所需要的算法和数据结构。

（3）编码和单元测试

在编码和单元测试阶段，程序员根据实际需要选取一种高级程序设计语言，把详细设计的结果翻译成用选定的语言书写的程序，并且仔细测试编写出的每一个模块。

（4）综合测试

综合测试阶段的主要任务是通过各种类型的测试及相应的调试，以发现功能、逻辑和实现上的缺陷，使软件达到预定的要求。

3. 维护时期

这个阶段的主要任务是进行各种修改，使系统能持久地满足用户的需要。维护阶段要

进行再定义和再开发，所不同的是在软件已经存在的基础上进行。

通常有 4 类维护活动：改正性维护，即诊断和改正在使用过程中发现的软件错误；适应性维护，即修改软件使之能适应环境的变化；完善性维护，即根据用户的新要求扩充功能和改进性能；预防性维护，即修改软件为将来的维护活动预先准备。

在软件工程中的每一个阶段完成后，为了确保活动的质量，必须进行评审。为了保证系统信息的完整性和软件使用的方便，还要有相应的文档资料。各阶段需要编写的文档与软件生命周期的关系如表 1.1 所示。

表 1.1　软件生命周期各阶段与文档编制的关系

阶段 / 文档	可行性研究	需求分析	概要设计	详细设计	编码和单元测试	综合测试
可行性研究报告	√					
项目开发计划书	√	√				
需求规格说明书		√				
测试计划		√	√	√		
概要设计说明书			√			
详细设计说明书				√		
数据库设计说明书			√	√		
用户手册		√	√	√	√	
操作手册			√	√	√	
测试分析报告						√
开发进度月报	√	√	√	√	√	√
项目开发总结						√

每个软件文档最终要回答如下问题。

(1) 为什么要开发与维护软件，即回答“为什么(why)”。

(2) 最终目标要满足哪些需求，即回答“做什么(what)”。

(3) 功能需求应如何实现，即回答“怎么做(how)”。

(4) 开发与维护软件计划由谁来完成，即回答“谁来做(who)”。

(5) 工作时间如何安排，即回答“何时做(when)”。

(6) 工作在什么环境中进行，所需信息从哪里来，即回答“何处做(where)”。

表 1.1 中的文档要回答哪些问题，请参考表 1.2。

表 1.2　软件文档所回答的问题

问题 / 文档	为什么	做什么	怎么做	谁来做	何时做	何处做
可行性研究报告	√	√				
项目开发计划书		√		√	√	
需求规格说明书		√				√
测试计划			√	√	√	
概要设计说明书			√			
详细设计说明书			√			

续表

问题 / 文档	为什么	做什么	怎么做	谁来做	何时做	何处做
数据库设计说明书			√			
用户手册			√			
操作手册			√			
测试分析报告		√				
开发进度月报		√			√	
项目开发总结		√				

1.2.2 能力目标

理解软件生命周期每个阶段所完成的核心任务。

1.2.3 任务驱动

任务 1：试想："在软件开发中，编写出正确的程序即可完成任务。"这句话正确吗？请说明理由。

任务 2：为什么要把软件生命周期划分成若干个阶段？

1.2.4 实践环节

假设你是一家软件公司的项目经理(PM)，当让你手下的软件工程师们撰写详细设计文档时，有人说："写这些文档没有用，我们直接写代码吧！"怎么反驳他？

1.3 常用的软件开发模型

1.3.1 核心知识

软件开发模型是贯穿整个软件生命周期(开发、运行和维护)所实施的全部工作和任务的结构框架，它描述了软件开发过程各阶段之间的关系。目前，常见的软件开发模型有瀑布模型、快速原型模型、增量模型、螺旋模型、喷泉模型、第四代技术过程模型等。下面简单介绍瀑布模型与快速原型模型。

1. 瀑布模型

瀑布模型即生存周期模型，其核心思想是按工序将问题简化，将功能的实现与设计分开，便于分工协作，即采用结构化的分析与设计方法将逻辑实现与物理实现分开。瀑布模型规定了软件生命周期各阶段的工作自上而下、相互衔接的固定次序，如同瀑布流水逐级下落。采用瀑布模型的软件过程如图 1.1 所示。

瀑布模型是最早出现的软件开发模型，在软件工程中占有极其重要的地位，它提供了软件开发的基本框架。瀑布模型的本质是一次通过，即每次活动只执行一次，最后得到软件产品，也称为"线性顺序模型"或者"传统生命周期"。其过程是从上一项活动接收该项活动的工作对象作为输入，利用这一输入实施该项活动应完成的内容，给出该项活动的工作成果，

并作为输出传给下一项活动。同时评审该项活动的实施，若确认，则继续下一项活动；否则返回前面，甚至更前面的活动。

瀑布模型有利于大型软件开发过程中人员的组织及管理，有利于软件开发方法和工具的研究与使用，从而提高了大型软件项目开发的质量和效率。然而软件开发的实践表明，上述各项活动之间并非完全是自上而下且呈线性图式的，因此瀑布模型存在严重的缺陷。

瀑布模型软件开发方法适合在软件需求比较明确、开发技术比较成熟、工程管理比较严格的场合下使用，如二次开发或升级型的项目。

2. 快速原型模型

快速原型模型的第一步是快速建立一个能满足用户基本需求的原型系统，使用户通过这个原型初步表达出自己的需求，并通过反复修改、完善，逐步靠近用户的全部需求，最终形成一个完全满足用户需求的新系统。

通过建立原型，可以更好地和客户进行沟通，澄清一些模糊需求，并且对需求的变化有较强的适应能力。原型模型可以减少技术、应用的风险，缩短开发时间，减少费用，提高生产率，通过实际运行原型，提供了用户直接评价系统的方法，促使用户主动参与开发活动，加强了信息的反馈，促进了各类人员的协调交流，减少误解，能够适应需求的变化，最终有效提高软件系统的质量。

快速原型模型软件开发方法适用于软件需求不明确的情况。从图 1.2 可以看出，快速原型模型的开发步骤如下。

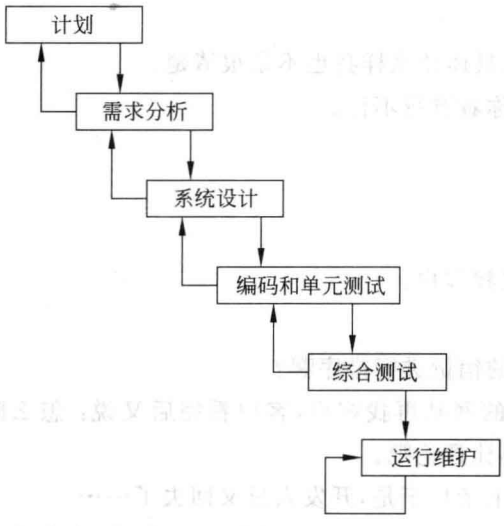

图 1.1　瀑布模型

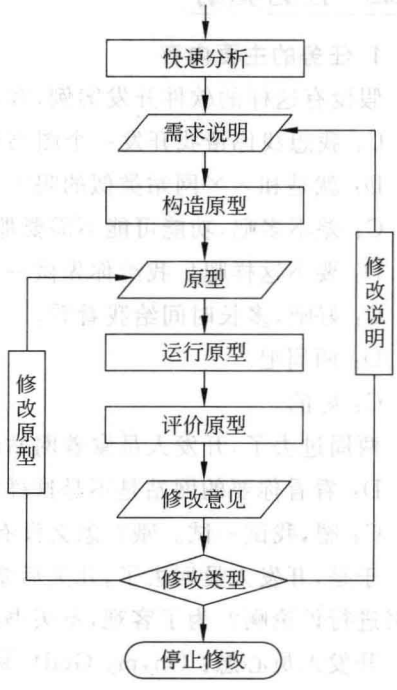

图 1.2　快速原型模型

(1) 快速分析

在分析人员与用户的密切配合下，迅速确定系统的基本需求，根据原型所要体现的特征

描述基本需求以满足开发原型的需要。

(2) 构造原型

在快速分析的基础上,根据基本需求说明尽快实现一个可行的系统。这里要求具有强有力的软件工具的支持,并忽略最终系统在某些细节上的要求,如安全性、坚固性、异常处理等,主要考虑原型系统能够充分反映所要评价的特性,而暂时删除一切次要内容。

(3) 运行原型

这是发现问题、消除误解、开发者与用户充分协调的一个步骤。

(4) 评价原型

在运行原型的基础上,考核评价原型的特性,分析运行效果是否满足用户的愿望,消除和纠正过去交互中的误解与分析中的错误,增添新的要求,并满足因环境变化或用户的新想法引起的系统变动需求,提出全面的修改意见。

(5) 修改类型

根据评价原型的活动结果进行修改。若原型未满足需求说明的要求,就说明对需求说明存在不一致的理解或实现方案不够合理,此时应根据明确的要求迅速修改原型。

1.3.2 能力目标

理解常用的软件开发模型的特点与开发流程。

1.3.3 任务驱动

1. 任务的主要内容

假设有这样的软件开发案例,客户(C)与开发人员(D)对话如下。

C:我想找你帮我开发一个图书销售网站,可以吗?

D:就是和××网站类似的吗?

C:差不多吧,功能可能不需要那么强大,具体什么样我也不是很清楚。

D:要不这样吧!我给你先做一个雏形,你看看行不行。

C:好吧,多长时间给我看看。

D:两周吧。

C:好的。

两周过去了,开发人员拿着网站的雏形来找客户。

D:看看你要的网站是不是这样的?

C:嗯,我试一试。哦?怎么没有对图书的销量进行排序啊?

于是,开发人员回去了,几天后拿着改好的网站再找客户,客户看完后又说:怎么随便对书进行评论啊?为了客观,不买书的会员不让评论吧。

开发人员心想:Oh,my God!顾客就是上帝!于是,开发人员又回去了……

就这样,客户反反复复地提出新的要求,开发人员不断地修改网站。最后,客户终于满意了图书销售网站的功能。

请问该图书销售网站的开发过程是属于哪种开发模型?

2. 任务分析

从上述案例分析得知：

(1) 客户并不是很清楚自己网站的需求；

(2) 开发人员快速开发出网站的雏形；

(3) 客户运行雏形，提出新的需求；

(4) 开发人员修改网站的雏形；

(5) 再运行雏形、再提出需求、再修改雏形，一直到客户满意为止。

因此，该案例的开发过程应属于快速原型模型。

3. 任务小结或知识扩展

瀑布模型中每个阶段的结果是一个或多个经过核准的文件。直到上一个阶段完成，下一阶段才能启动。在实际过程中，这些阶段经常是重叠的、彼此间有信息交换的。在设计阶段，需求中的问题被发现；在编程阶段，设计问题被发现，以此类推。软件过程不是一个简单的线性模型，它包括开发活动的多个反复。因此它的缺点是：委托事项需要在过程的早期阶段清晰给出，响应用户需求的变更比较困难。快速原型模型就是为了克服瀑布模型的缺点而提出来的。

总之，当需求不太明确的时候，应采用快速原型模型；当需求比较明确的时候，应采用瀑布模型。由于瀑布模型反映了工程的实际情况，所以在大型系统工程项目中，软件开发仍采用瀑布模型。

4. 任务的参考答案

【答案】 快速原型模型

1.3.4 实践环节

分析如下软件开发案例，判断该案例的开发过程是属于哪种软件开发模型。

某个老师(T)想要考查一个同学(S)的学习情况和技术水平，于是交给该学生一个任务。

1) 两个人的对话

T：我想要一个单词统计软件，统计功能包括：单词总数、单个单词重复出现的个数，并按字典顺序排序。你能做这样一个软件吗？

S：就是这样的软件吗？如，一篇文章内容为：How are you? Fine, thank you. 统计结果为：单词总数 6，are 1 Fine 1 How 1 thank 1 you 2。

T：嗯，可以。

S：这个我上网查查相关资料，应该没有问题，很简单的。

T：好的，给你 10 天时间，两天之后你再来一趟，讲一下你的工作进度。

2) 工作清单

这位同学非常明白老师的意图，回去后想了一下，列出了一个清单，工作清单如下。

(1) 功能

① 统计单词总数。

② 统计单个单词的数目。

(2) 其他说明

① 界面尽量简洁,容易操作。

② 处理速度尽量快。

③ 开发文档尽量齐全。

(3) 开发工具

Eclipse 3.7

(4) 开发环境

普通 PC

Windows 7/Windows XP 系统

JDK1.5

(5) 工作量

① 研究一下文档(存放文章内容)文件的格式。

② 设计一个解析器类,解析这些文件格式。

③ 设计一个文档类,实现读取功能。

④ 设计一个统计类,实现统计功能。

⑤ 设计一个视图类,实现按要求显示功能。

3) 案例的实际情况

一切顺利,学生S按期交付了软件,经过一两周的检查、试用、修改、完善后,该软件在老师那里成为得心应手的工具。

1.4 小　　结

生命周期方法学把软件生命周期从时间角度划分为软件定义、软件开发和软件维护三个时期,每个时期又划分为若干个阶段。每个阶段结束之前都从技术和管理两个角度进行严格的审查。

瀑布模型有利于大型软件开发过程中人员的组织及管理,也有利于提高大型软件项目开发的质量和效率。然而实践表明,软件开发的各项活动之间并非完全是自上而下且呈线性图式的,因此瀑布模型存在严重的缺陷。

快速原型模型就是为了克服瀑布模型的缺点而提出来的。它通过快速建立一个能满足用户基本需求的原型系统,使用户通过这个原型初步表达出自己的要求,并通过反复修改、完善,逐步靠近用户的全部需求,最终形成一个完全满足用户需求的新系统。

习　题　1

一、单项选择题

1. 软件是(　　)。

A. 处理对象和处理规则的描述　　B. 程序

C. 程序、数据及文档　　D. 计算机系统