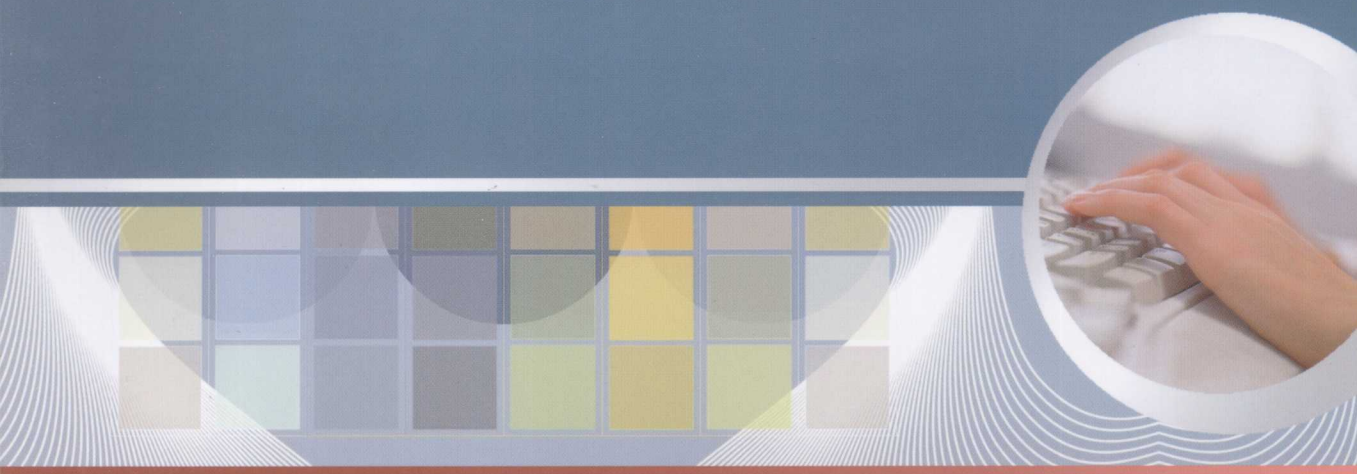
实例讲解
实训强化
培养技能
面向就业

全国高等职业教育计算机类规划教材 · 实例与实训教程系列

软件开发过程与项目管理

◎ 杨学瑜　高立军　编著



- 软件开发文档规范、齐备
- 软件架构设计先进，符合面向对象的理念
- 软件工程的概念、理论和方法融入具体项目
- 程序实例来自实际项目，增加实例剖析内容

電子工業出版社
PUBLISHING HOUSE OF ELECTRONICS INDUSTRY
http://www.phei.com.cn

全国高等职业教育计算机类规划教材·实例与实训教程系列

软件开发过程与项目管理

杨学瑜　高立军　编著

電子工業出版社

Publishing House of Electronics Industry

北京·BEIJING

内 容 简 介

本书通过基于用 JSP 和 Java 编写的“项目工单管理系统”应用程序，把软件工程的概念、理论、方法和技术融入到具体项目中，让学生在直观的事例中体会和理解知识，并安排一定的实训学时，按照项目管理方法，让学生分组进行开发。在软件开发中通过版本控制软件，熟悉软件版本控制的原理和方法；编写测试用例、实施测试，编写测试报告，体会测试的全过程。其内容包括需求分析、架构设计、详细设计、编码和调试、测试、安装部署及项目管理。

本书适合高职高专院校、成人高校及本科院校的二级职业技术学院教学用书，还可供本科院校、IT 从业人员及爱好者参考。

图书在版编目(CIP)数据

软件开发过程与项目管理 / 杨学瑜，高立军编著. —北京：电子工业出版社，2008.2
（全国高等职业教育计算机类规划教材 • 实例与实训教程系列）
ISBN 978-7-121-05537-9

Ⅰ. 软… Ⅱ. ①杨…②高… Ⅲ. 软件开发－项目管理－高等学校：技术学校－教材 Ⅳ. TP311.52

中国版本图书馆 CIP 数据核字（2007）第 189397 号

策划编辑：程超群
责任编辑：韩玲玲
印　　刷：北京牛山世兴印刷厂
装　　订：
出版发行：电子工业出版社
　　北京市海淀区万寿路 173 信箱　邮编 100036
开　　本：787×1 092　1/16　印张：13.25　字数：339 千字
印　　次：2008 年 2 月第 1 次印刷
印　　数：4 000 册　　定价：22.00 元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至 zlts@phei.com.cn，盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线：（010）88258888。

序

20世纪90年代以来，以计算机和通信技术为推动力的信息产业在我国获得前所未有的发展，全国各企事业单位对信息技术人才求贤若渴，高等教育计算机及相关专业毕业生供不应求。随后几年，我国各高等院校、众多培训机构相继开设计算机及相关专业，积极扩大招生规模，不久即出现了计算机及相关专业毕业生供大于求的局面。纵观近十年的就业市场变化，计算机专业毕业生经历了“一夜成名、求之不得”的宠幸，也遭遇了“千呼百应，尽失风流”的冷落。

这个时代深深地镌刻着信息的烙印，这个时代是信息技术人才尽情展示才能的舞台。目前我国的劳动力市场，求职人数过剩，但满足企业要求的专业人才又很稀缺。这种结构性的人才市场供求矛盾是我国高等教育亟待解决的问题，更是“以人为本，面向人人”为目标的职业教育不可推卸的责任。

电子工业出版社，作为我国出版职业教育教材最早的出版社之一，是计算机及相关专业高等职业教材重要的出版基地。多年来，我们一直在教材领域为战斗在职业教育第一线的广大职业院校教育工作者贡献着我们的力量，积累了丰富的职业教材出版经验。目前，计算机专业高等教育正处于发展中的关键时期，我们有义务、有能力协同全国各高等职业院校，共同探寻适合社会发展需要的人才培养模式，建设满足高等职业教育需求的教学资源——这是我们出版“全国高等职业教育计算机专业规划教材——实例与实训教程系列”的初衷。

关于本系列教材的出版，我们力求做到以下几点：

（1）面向社会人才市场需求，以培养学生技能为目标。工学结合、校企结合是职业教育发展的客观要求，面向就业是职业教育的根本落脚点。本系列教材内容体系的制定是广大高职教育专家、一线高职教师共同智慧的结晶。我们力求教材内容丰富而不臃肿、精简而不残缺，实用为主、够用为度。

（2）面向高职学校教师，以方便教学为宗旨。针对每个课程的教学特点和授课方法，我们为其配备相应的实训指导、习题解答、电子教案、教学素材、阅读资料、程序源代码、电子课件、网站支持等一系列教学资源，广大教师均可从华信教育资源网（www.huaxin.edu.cn）免费获得。

（3）面向高职学校学生，以易学、乐学为标准。以实例讲述理论、以项目驱动教学是本系列教材的显著特色。这符合现阶段我国高职学生的认知规律，能够提高他们的学习兴趣，增强他们的学习效果。

这是一个崭新的开始，但永远没有尽头。高等职业教育教材的建设离不开广大职业教育工作者的支持，尤其离不开众多高等职业院校教师的支持。我们诚挚欢迎致力于职业教育事业发展的有识之士、致力于高等职业教材建设的有才之士加入到我们的队伍中来，多批评，勤点拨，广结友，共繁荣，为我国高等职业教育的发展贡献我们最大的力量！

电子工业出版社高等职业教育分社

前 言

软件工程是研究软件开发与软件项目管理的一门工程科学，是计算机应用与软件工程等相关专业的主干课程，也是软件开发人员、软件销售工程师、软件高层决策者必不可少的专门知识，理论性较强。而软件开发与项目管理是与软件工程类似的一门课程，侧重于理论的具体应用。本书作者先在软件企业从事软件项目开发与项目管理工作，积累了大量的技术和管理经验；之后到高职学院从事教学与软件项目开发工作，把实际工程项目引入到教学中去。目前，许多高职高专院校的计算机类专业都在开设软件工程课程。可是在实际的教学过程中，作者发现许多软件工程教材因为理论性强、文字多、内容枯燥等原因而无法吸引学生。那么如何才能把软件开发过程和软件项目管理的知识传递变得生动有趣又有实用意义呢？经过多次的教学实践，作者提出以软件项目应用为主线的软件工程教学方法，即采用业界流行的项目管理、软件分析设计、编码和测试工具进行软件项目的管理、设计、编码和测试，同时使用通用的版本控制工具进行版本控制。按照这种思路，开发了这本教材，并依据 IT 企业的软件开发流程，面向工程实践安排了书中的章节。

第 1 章介绍了软件开发过程的基本概念、方法和技术；第 2 章介绍了软件开发项目管理的基本知识和方法，包括软件项目进度、质量和成本管理知识及项目管理软件的使用；第 3 章介绍了软件需求获取与分析的概念和方法，包括使用用例图进行需求建模的简单应用；第 4 章阐述了软件架构设计的概念及任务，以及在工单管理系统架构设计中的应用；第 5 章介绍了软件详细设计的基础知识和设计工具的简单用法，阐述了面向对象的基本概念和 MVC 设计模式，并以工单类别管理模块为例简述了详细设计内容；第 6 章论述了软件开发语言及工具选择的原则，介绍了编码规范、编程风格、开发环境的搭建及程序调试方法，以工单类别管理系统为例，讲解程序调用关系及代码含义；第 7 章介绍了软件测试的基本知识和测试工具的简单使用，阐述了测试用例及测试报告的编写规范及方法；第 8 章介绍了软件部署、维护的基本知识，以及软件部署的方法；第 9 章提供了“项目工单管理系统”软件开发的详细开发任务书，包括编码和测试，目的是学以致用，加深对软件开发过程及项目理论、方法和技术的理解。

本书由杨学瑜、高立军编著，具体分工是：第 1 章、第 3 章、第 4 章、第 5 章、第 6 章和第 9 章由杨学瑜撰写；第 2 章、第 7 章和第 8 章由高立军撰写。

本书在撰写和出版过程中，得到了电子工业出版社的大力支持，在此特致谢意。

由于水平有限，时间紧张，书中错误之处在所难免，敬请业界专家、学者批评指正。

作 者

目　录

第1章 软件开发过程概述

内容提要及学习要求

1. 软件及软件开发的概念；
2. 软件开发过程与方法；
3. 软件开发过程的工程化理念；
4. UML 的含义及其简单使用。

了解软件开发过程的 6 个阶段，掌握软件开发的工程化思想，了解 UML 含义并能够使用它进行简单的分析和设计。

1.1 软件开发概述

1.1.1 软件的概念

如果没有软件（Software），计算机什么也不会做。那么什么是计算机软件呢？计算机软件（Computer Software）是指计算机系统中的程序、数据及其文档。为了弄清软件的概念，应先了解程序（Program）、数据（Data）及文档（Document）的概念。

1. 程序

程序是计算机为完成特定任务而执行的指令的有序集合，通常用某种程序设计语言编写，运行于某种目标体系结构上。打个比方，一个程序就像一个用汉语（程序设计语言）写下的红焖大虾菜谱（程序），用于指导懂汉语的人（体系结构）来做这个菜。通常，计算机程序要经过编译和链接而成为一种人们不易理解而计算机理解的格式（如二进制格式），然后运行。未经编译就可运行的程序通常称之为脚本程序。

在实际应用中，站在不同的角度，对程序概念的理解也有差异：

- 面向过程的程序=算法+数据结构；
- 面向对象的程序=对象+消息；
- 面向组件的程序=组件+架构。

程序设计的最终结果是软件。

2. 数据

程序已经被定义了，那么如何定义数据呢？数据可以定义为被程序处理的信息。但当我们考虑整个计算机系统时，有时程序和数据的区别就不是那么明显了。数据可以是一个有待执行的程序（参见脚本编程语言），程序可以编写成为能够编写其他程序的程序，所以有人甚至断言程序和数据没有区别。

3. 文档

至于文档（Document），通常可以这样理解：它指的是一些记录的数据和数据媒体，具

有固定不变的形式，可被人和计算机阅读。文档和计算机程序及数据共同构成了能完成特定功能的计算机软件（有人把源程序也当做文档的一部分）。

我们知道，硬件产品和产品资料在整个生产过程中都是有形可见的，软件生产则有很大的不同，文档本身就是软件产品。没有文档的软件不称其为软件，更谈不上软件产品。同时，软件文档的编制（Documentation）在软件开发工作中占有突出的地位和相当的工作量。高效率、高质量地开发、发布、管理和维护文档，对于转让、变更、修正、扩充和使用文档，以及充分发挥软件产品的效益有着重要意义。

然而，在实际工作中，文档在编制和使用中存在着许多问题，有待于解决。软件开发人员中较普遍地存在着对编制文档不感兴趣的现象。从用户方面看，他们又常常抱怨：文档售价太高、文档不够完整、文档编写得不好、文档已经陈旧或是文档太多，难于使用等等。究竟应该怎样要求它、文档应该写哪些、说明什么问题、起什么作用等，本章后文将给出简要的介绍。

1.1.2 编程与软件开发

1. 编程

编写程序，简称编程，是以下步骤的一个往复过程：编写新的源代码，测试、分析和提高新编写的代码以找出语法和语义错误。从事这种工作的人称做程序设计员。由于计算机的飞速发展，编程的要求和种类也日趋多样，由此产生了不同种类的程序设计员，每一种都有更细致的分工和任务，如软件工程师、架构设计师和系统分析师。

2. 软件开发

狭义上讲，软件开发就是长时间的编程过程；而在广义上，软件开发主要是计划、分析、设计、编码、测试和维护的全过程活动。

（1）计划。计划即对所要解决的问题进行总体定义，包括了解用户的要求及现实环境，从技术、经济和社会因素等 3 个方面研究并论证本软件项目的可行性，编写可行性研究报告，探讨解决问题的方案，并对可供使用的资源（如计算机硬件、系统软件、人力等）成本、可取得的效益和开发进度做出估计，制订完成开发任务的实施计划。

（2）分析。软件需求分析就是回答软件做什么的问题。它是一个对用户的需求进行去粗取精、去伪存真、正确理解，然后用软件工程开发语言（形式功能规约，即需求规格说明书）表达出来的过程。本阶段的基本任务是和用户一起确定要解决的问题，建立软件的逻辑模型，编写《用户需求说明书》和《需求规格说明书》文档并最终得到用户的认可（前者是面向用户的，作为软件开发合同的附件；后者是面向设计人员的，作为软件设计的输入文件）。需求分析的主要方法包括面向过程（结构化）的分析方法和面向对象的分析方法。

（3）设计。实际上软件设计的主要任务就是将软件分解成模块。模块是指能实现某个功能的数据和程序说明、可执行程序的程序单元，可以是一个函数、过程、子程序、一段带有程序说明的独立的程序和数据，也可以是可组合、可分解和可更换的功能单元或组件。模块分解后，接下来就要进行模块设计。

软件设计可以分为架构（概要）设计和详细设计两个阶段。架构设计就是结构设计，其主要目标就是给出软件的模块结构以及模块之间的接口和关联关系，用软件结构图表示。在详细设计过程中，面向过程设计的首要任务就是设计模块的程序流程、算法和数据结构，次要任务就是设计数据库；而面向对象设计的主要任务是用例实现设计、类设计及数据存储设计。

（4）编码。软件编码是指把软件设计转换成计算机可以接受的程序，即写成以某一程序设计语言表示的“源程序清单”。充分了解软件开发语言、工具的特性和编程风格，有助于开发工具的选择以及保证软件产品的开发质量。

当前，软件开发中除在专用场合，已经很少有人使用 20 世纪 80 年代的高级语言了，取而代之的是面向对象的开发语言。面向对象的开发语言和开发环境大都合为一体，大大提高了开发的速度和效率。

（5）测试。软件测试的目的是以较小的代价发现尽可能多的错误。要实现这个目标的关键在于设计一套出色的测试用例（测试数据和预期的输出结果组成了测试用例）。如何才能设计出一套出色的测试用例，关键在于理解测试方法。不同的测试方法有不同的测试用例设计方法，两种常用的测试方法是白盒法和黑盒法。白盒法的测试对象是源程序，依据程序内部的逻辑结构来发现软件的编程错误、结构错误和数据错误（结构错误包括逻辑、数据流、初始化等错误）。其用例设计的关键是以较少的用例覆盖尽可能多的内部程序逻辑结果。黑盒法依据的是软件的功能或软件行为描述，以发现软件的接口、功能和结构错误。其中，接口错误包括内部/外部接口、资源管理、集成化以及系统错误。同样黑盒法用例设计的关键也是以较少的用例覆盖模块输出和输入接口。

（6）维护。维护是指在已完成对软件的研制（分析、设计、编码和测试）工作并交付使用以后，对软件产品所进行的一些软件工程的活动，即根据软件运行的情况，对软件进行适当修改，以适应新的要求，以及纠正运行中发现的错误，并编写软件问题报告、软件修改报告。

一个中等规模的软件，如果研制阶段需要 1～2 年的时间，在它投入使用以后，其运行或工作时间可能持续 5～10 年。那么它的维护阶段也是运行的这 5～10 年期间。在这段时间，人们几乎需要着手解决研制阶段所遇到的各种问题，同时还要解决某些维护工作本身特有的问题。做好软件维护工作，不仅能排除障碍，使软件正常工作，而且还可以使它扩展功能，提高性能，为用户带来明显的经济效益。然而遗憾的是，现实中对软件维护工作的重视往往远不如对软件研制工作的重视。事实上，和软件研制工作相比，软件维护的工作量和成本都要大得多。

在实际开发过程中，软件开发并不是从第一步进行到最后一步，而是在任何阶段、在进入下一阶段前一般都有一步或几步的回溯。例如，在测试过程中出现的问题可能要求修改设计，用户可能会提出一些需要来修改需求说明书等。

1.1.3 软件开发过程与方法

1. 软件开发模式

（1）统一软件过程（RUP）。RUP 是 Rational Unified Process 的缩写，翻译成中文就是“统一软件过程”。RUP 是一套软件工程方法，主要由 Ivar Jacobson 的 The Objectory Approch 和 The Rational Approch 发展而来；同时，它又是文档化的软件工程产品，所有 RUP 的实施细节及方法介绍均以 Web 文档的方式集成在一张光盘上，由前 Rational 公司（目前已被 IBM 公司收购）开发、维护并销售，当前版本是 5.0。RUP 又是一套软件工程方法的框架，各个组织均可根据自身的实际情况及项目规模对 RUP 进行裁剪和修改，以制定出合乎需要的软件工程过程。

RUP 是一个基于 6 个最佳开发实践的流程定义产品。这 6 个最佳开发实践是：

- 迭代始开发；
- 需求管理；
- 基于组建的体系架构；
- 可视化建模；
- 持续的质量管理；
- 配置管理。

那么在软件开发过程中，RUP 如何来实现这 6 个最佳开发实践呢？

① 首先，把软件开发过程看成是多次迭代开发的过程，并且把迭代开发分成以下 4 个阶段：

- Inception Phase（开始阶段），定义出项目目标和范围；
- Elaboration Phase（细化阶段），制定计划、定义项目基线、确定系统的体系架构；
- Construction Phase（开发阶段），主要是编码、单元测试工作，是人工最密集的阶段（这个时候，虽然允许有小的需求加入进来，但是应该尽量避免大的需求变动）；
- Transition Phase（发布阶段），将产品提交给用户使用（包括相关的培训等内容）。

【注意】 每个阶段由若干次迭代组成。

② 其次，定义出一次迭代开发所要遵循的 9 个规则（Disciplines）：

- 业务建模（Business Modeling）；
- 需求（Requirements）；
- 分析和设计（Analysis & Design）；
- 实现（Implementation）；
- 测试（Test）；
- 部署（Deployment）；
- 项目管理（Project Management）；
- 配置与变更管理（Configuration & Change Management）；
- 环境（Environment）。

其中前 6 个称为核心工程工作流（Core Engineering Workflows），后 3 个称为辅助工作流（Supporting Workflows）。

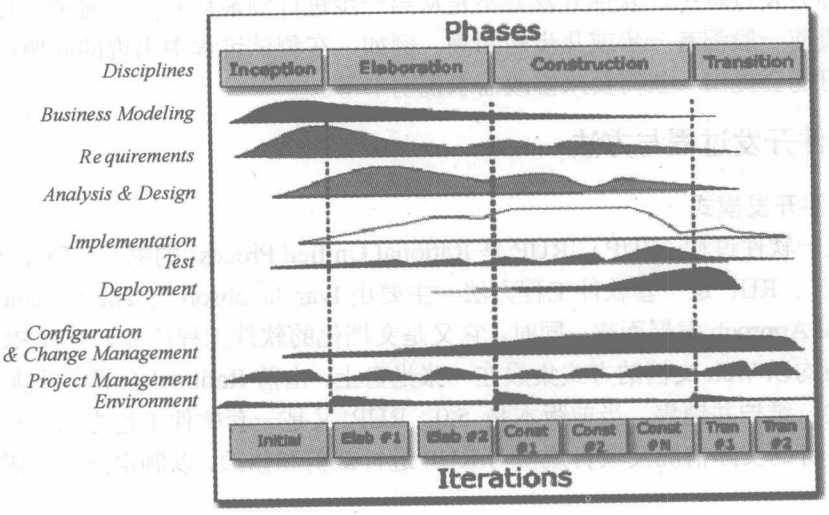


图 1.1 迭代开发规则

在每次迭代中，都要经历所有的规则。其实，RUP 所定义的 9 个规则，跟瀑布式开发过程是相类似的，即需求→分析、设计→开发→测试→部署。

RUP 的本质特点是：

- RUP 是风险驱动的、基于用例技术的、以架构为中心的、迭代的、可配置的软件开发流程；
- 可以针对 RUP 所规定出的流程，进行客户化定制，定制出适合自己组织的实用的软件流程。

（2）敏捷开发过程。敏捷开发（Agile Development）是一种以人为核心、迭代、循序渐进的开发方法。在敏捷开发中，软件项目的构建被切分成多个子项目，各个子项目的成果都经过测试，具备集成和可运行的特征。简言之，就是把一个大项目分为多个相互联系、但也可独立运行的小项目，并分别完成，在此过程中软件一直处于可使用状态。典型的敏捷开发过程包括 XP（Extreme Programming，极限编程）、FDD（特性驱动开发）、Scrum（英式橄榄球争球队）以及敏捷的统一过程（AUP）等。限于篇幅，本书只介绍 XP 开发。

XP 是由 Kent Beck 在 1996 年提出的。Kent Beck 在与 Ward Cunningham 共事时，就一直共同探索着新的软件开发方法，希望能使软件开发更加简单而有效。他仔细地观察和分析了各种简化软件开发的前提条件、可能性以及面临的困难，终于于 1996 年 3 月，在为 Daimler Chrysler 所做的一个项目中引入了新的软件开发观念——XP。

XP 的目标是在规定的时间内生产出满足客户需要的软件。当一个中小型软件开发团队（人数不超过 10 个）遇到用户需求不明确、变化快，而且在特定的时间内将面对一个相当难开发的系统时，可以采用敏捷开发过程。XP 体现了沟通（Communication）、简化（Simlicity）、反馈（Feedback）和勇气（Courage）四个价值。

XP 包括 12 个实践活动：

- 有计划的开发，通过结合使用优先级“故事”和技术估算，确定下一版本的功能；
- 小版本，以小的增量版本经常向客户发布软件；
- 隐喻，一个简单、共享的“故事”或描述，说明系统如何工作；
- 简单设计，通过保持代码简单从而保证设计简单；不断地在代码中寻找复杂点并且立刻进行移除；
- 测试驱动开发，用户编写测试内容以对“故事”进行测试；程序员编写测试内容来发现代码中的任何问题（在编写代码前先编写测试内容）；
- 重构，这是一项简化技术，用来移除代码中的重复内容和复杂之处；
- 结对编程，团队中的两个成员使用同一台计算机开发所有的代码，一个人编写代码或者驱动，另一个人同时审查代码的正确性和可理解性；
- 集体代码所有权，任何人都拥有所有的代码，这就意味着每个人都可以在任何时候变更任何代码；
- 持续集成，每天多次创建和集成系统，只要任何实现任务完成就要进行；
- 每周 40 个小时，程序员在疲劳时无法保证最高效率，所以连续两周加班是绝对不允许的；
- 现场客户，一名真实的客户全时工作于开发环境中，帮助定义系统、编写测试内容并回答问题；
- 编码标准，程序员采用一致的编码标准。

不适用于 XP 的场合：

- 不能接受 XP 文化的组织；
- 中大型（超过 10 个人）的团队；
- 重构会导致大量开销的应用；
- 需要很长的编译或测试周期的系统；
- 不太容易测试的应用；
- 人员异地分布的物理环境。

（3）两种开发过程的比较。每种软件方法产生的背景不同、特点不同，适用的领域亦不相同。那么对于软件项目来说，是否可以使用同一种软件开发过程来对不同类型、规模、复杂度的项目来进行开发呢？显然是不合理的。

对于 RUP 来说，首先，它过于理想化和理论化。RUP 是过程组件、方法以及技术的框架，可以将其应用于任何特定的软件项目，由用户自己限定 RUP 的使用范围。但对于各种类型的软件项目，RUP 并未给出具体的自身裁减及实施策略，总有些无依据可循的感觉。既可以说它能解决任何问题，也可以说它什么都不是。其次，RUP 从本质上来说还是一个强调设计和规范的软件方法。从这个角度来讲，RUP 与传统的瀑布模型没有太大的差别，它的灵活性较之敏捷方法还是相对较弱的。在一些小型软件项目、特别是不可预测的软件项目开发中，面临着各种紧急需求和时间压力，沿用 RUP 是很难应付自如的。但是另一方面，RUP 强调对知识的收集、整理和加工定义，强调在软件开发的时候要有好的体系结构，所以它还是很有利于知识的积累和共享的。

相比之下，敏捷方法（如 XP）则更为灵活。敏捷方法倡导尽早地、持续地交付有价值的软件满足用户需要，用交流沟通取代详尽的文档，强调团队的主动、自律、自我组织和自发管理。而 XP 也是以代码为核心的一种方法，这里有很多的东西是未知的，知识只存在于两个地方：开发者的头脑和最后的代码。对于项目管理者来说，他们会认为敏捷开发方法弱化了知识管理的概念，而实际上敏捷开发注重的是最有价值的知识的积累和沉淀。

如何灵活应对各种项目风险，如何最大化优先满足用户价值，如何能够有效控制项目开发过程，如何做好项目过程中的知识管理，是每一个软件项目管理者都需要深入思考的问题。RUP 的倡导者一直强调 RUP 裁剪，实际上不仅仅需要自身的裁剪，还需要学会融合，即在 RUP 裁剪的同时，适宜地融合敏捷开发的各种实践。不要认为 RUP 与 XP 是矛盾的，其实不然，它们具有不同的原理、不同的应用领域。在 RUP 中融合了 XP 技术时，才会得到过程的正确量，既满足了项目所有成员的需要，又解决了所有主要的项目风险问题。对于一个工作于高度信任环境中的小型项目团队，其中用户是团队的一部分，那么 XP 完全可以胜任；对于团队越来越分散，代码量越来越大，或者构架没有很好定义的情况，则需要做一些其他工作。在用户交互具有“契约”风格的项目中，仅有 XP 是不够的，RUP 是一个框架，可以从 RUP 出发，在必要时以一组更健壮的技术来扩展 XP。

在软件项目开发过程中，应该能够识别、分析不同软件项目的特点，采用相对适合的开发实践来适应软件开发过程，保证对软件开发的有效支持，以便能够创造出“足够好的”软件。而“足够好的”就是对进度、成本、质量之间的平衡，以及最大化满足客户需要的实现。

2．软件开发方法

（1）面向过程的方法。面向过程的开发方法包括面向过程分析、面向过程设计、面向过程编程、面向过程测试和面向过程维护。面向过程的方法又称结构化（或面向功能）方

法，习惯上称做结构化分析、结构化设计、结构化编程、结构化测试和结构化维护。结构化方法开始于20世纪60年代，成熟于70年代，80年代在软件行业得到广泛接受和使用，并一度成为早期占主导地位的软件构造与开发方法。目前在嵌入式软件开发中，还大量采用这种方法。

面向过程方法的特点是，程序的执行过程不由用户控制，完全由程序控制。面向过程的方法具有简单实用的优点，但维护困难。

（2）面向对象的方法。在软件开发过程中，使用者会不断地提出各种更改要求，即使在软件投入使用后，也常常需要对其做出修改。在用结构化开发的程序中，这种修改往往是很困难的，而且还会因为计划或考虑不周，不但旧错误没有得到彻底改正，又引入了新的错误；另一方面，在过去的程序开发中，代码的重用率很低，使得程序员的效率并不高。为提高软件系统的稳定性、可修改性和可重用性，人们在实践中逐渐创造出软件开发的一种新途径——面向对象的方法。

面向对象的方法开始于20世纪80年代，兴起于90年代，目前开始走向成熟。面向对象的开发方法包括面向对象分析、面向对象设计、面向对象编程、面向对象测试和面向对象维护。面向对象方法的特点是，程序的执行过程不由用户控制，完全由程序交互控制。面向对象的方法具有易于维护、代码可复用等优点，但较难掌握。

面向对象方法的出发点和基本原则是尽可能模拟人类习惯的思维方式，使软件开发的方法与过程尽可能接近人类认识世界、解决问题的方法与过程。由于客观世界的问题都是由客观世界中的实体及实体相互间的关系构成的，因此我们把客观世界中的实体抽象为对象（Object）。持面向对象观点的程序员认为计算机程序的结构应该与所要解决的问题一致，而不是与某种分析或开发方法保持一致。他们的经验表明，对任何软件系统而言，其中最稳定的成分往往是其相应问题论域（Problem Domain）中的成分。

通俗地讲，对象指的是一个独立的、异步的、并发的实体，它能“知道一些事情”（即存储数据），“做一些工作”（即封装服务），并“与其他对象协同工作”（通过交换消息），从而完成系统的所有功能。

因为所要解决的问题具有特殊性，所以对象是不固定的，一个雇员可以作为一个对象，一家公司也可以作为一个对象。到底应该把什么抽象为对象，由所要解决的问题决定。

面向对象的方法具有下述四个要点。

- 认为客观世界是由各种对象组成的，任何事物都是对象，复杂的对象可以由比较简单的对象以某种方式组合而成。按照这种观点，可以认为整个世界就是一个最复杂的对象。因此，面向对象的软件系统是由对象组成的，软件中的任何元素都是对象，复杂的软件对象由比较简单的对象组合而成。
- 把所有对象都划分成各种对象类（简称为类 Class），每个对象类都定义了一组数据和一组方法，数据用于表示对象的静态属性，是对象的状态信息。因此，每当建立该对象类的一个新实例时，就按照类中对数据的定义为这个新对象生成一组专用的数据，以便描述该对象独特的属性值。

例如，荧光屏上不同位置显示的半径不同的几个圆，虽然都是 Circle 类的对象，但是各自都有自己专用的数据，以便记录各自的圆心位置、半径等等。

类中定义的方法，是允许施加于该类对象上的操作，是该类所有对象共享的，并不需要为每个对象都复制操作的代码。

- 按照子类（或称为派生类）与父类（或称为基类）的关系，把若干个对象类组成一个层次结构的系统（也称为类等级）。
- 对象彼此之间仅能通过传递消息互相联系。

（3）两种方法的比较。结构化方法首先关心的是功能，强调以模块（即过程）为中心，采用模块化、自顶向下、逐步求精的设计过程。系统是实现模块功能的函数和过程的集合，结构清晰、可读性好，是提高软件开发质量的一种有效手段。

结构化设计从系统的功能入手，按照工程标准和严格规范将系统分解为若干功能模块。然而，由于用户的需求和软、硬件技术的不断发展变化，作为系统基本成分的功能模块很容易受到影响，局部修改甚至会引起系统的根本性变化，开发过程前期入手快而后期频繁改动的现象比较常见。

面向对象方法则从所处理的数据入手，以数据为中心来描述系统。数据相对于功能而言，具有更强的稳定性，这样设计出的系统模型往往能较好地映射问题域模型。对象、类、封装性、继承性和多态性等概念的引入使用，显然令面向对象的设计方法具有一定的优势，能为生产可重用的软件构件和解决软件的复杂性问题提供一条有效的途径。

面向对象的设计过程通过建立一些类以及它们之间的关系来解决实际问题，这就需要对问题域中的对象作整体分析。类和类间关系的设计要求较高，否则设计出的并不是真正意义上的面向对象的软件系统，而只是一些类的堆砌而已，不能体现出面向对象设计方法的优势之处。

同时，系统的分析设计是一个注重实践的领域，不仅仅依赖于一整套核心的概念与原理。要想设计出一个成功的系统来，还需要相应的语言、工具和技术的有力支持。在这方面，经过多年的实践和发展，适应结构化方法的技术和开发环境已经相当成熟稳定。而对面向对象方法而言，虽然近期涌现了大量的新工具和新技术，但仍有待于不断的完善和改进，特别是面向对象的数据库技术。

1.2 软件开发过程的工程化理念

1.2.1 软件危机

1. 概况

20 世纪 60 年代以前，计算机刚刚投入实际使用，这时的软件设计往往只是为了一个特定的应用而在指定的计算机上设计和编制，采用密切依赖于计算机的机器代码或汇编语言；软件的规模比较小，文档资料通常也不存在，很少使用系统化的开发方法，设计软件往往等同于编制程序，基本上是个人设计、个人使用、个人操作、自给自足的私人化的软件生产方式。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。高级语言的出现、操作系统的发展引起了计算机应用方式的变化；大量的数据处理导致第一代数据库管理系统的诞生。此时，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、个人使用的方式已不能满足要求，迫切需要改变软件生产方式，提高软件生产率——软件危机开始爆发。

2. 现象

软件危机主要表现在以下方面。