第 1 章

软件工程的内容与方法

本章导读

本章首先对软件、软件工程、软件工程学科体系、软件工程课程进行了定义与解释，然后提出“面向过程方法、面向对象方法、面向元数据方法、形式化方法”的软件工程方法论，以及“面向流程分析、面向元数据设计、面向对象实现、面向功能测试、面向过程管理”的“五个面向”软件工程实践论，该方法论与实践论不但适用于信息系统的开发，也适用于其他软件系统的开发。本章最后简要介绍了软件支持过程、软件过程改进、软件企业文化、信息系统的定义与案例分析。因此，本章是软件工程课程的绪论。表 1-1 列出了读者在本章学习中要了解、理解和关注的主要内容。

表 1-1　本章对读者的要求

要　求	具 体 内 容
了　解	（1）软件与软件危机 （2）软件工程的定义与作用 （3）软件工程研究的内容
理　解	（1）软件工程方法论 （2）软件工程实践论 （3）CMMI、ISO 9001、微软企业文化、敏捷文化现象
关　注	（1）软件工程的最新发展 （2）CASE 工具与软件工程环境 （3）软件业务基础平台 （4）面向方面方法和面向代理方法

1.1　软件的定义

1．计算机硬件与软件

计算机（Computer）由硬件（Hardware）和软件（Software）组成，硬件是看得见、摸得着的电子机械设备，如机箱、主板、硬盘、光盘、U 盘、电源、显示器、键盘、鼠标、打印机、电缆等。硬件是软件的载体，软件是依附在硬件上面的程序、数据和文档的集合，是指挥控制计算机系统（包括硬件系统和软件系统）工作的神经中枢。如果将硬件比作人的身体，那么软件就相当于人的神经中枢和知识才能。软件的分类比较复杂。分类方法不同，内容也不同。表 1-2 从 5 个不同角度对软件进行了分类。

表 1-2　软件的分类

序　号	分 类 方 法	软 件 内 容
1	按功能分类	（1）系统软件（如操作系统） （2）支撑软件（如数据库管理系统、CASE 工具系统） （3）应用软件（如信息系统）
2	按规模分类	（1）小型软件 （2）中型软件 （3）大型软件
3	按工作方式分类	（1）实时软件 （2）分时软件 （3）交互式软件 （4）批处理软件
4	按服务对象分类	（1）项目软件（为用户定制） （2）产品软件（面向特定的客户群开发）
5	按销售方式分类	（1）订单软件（已签订合同） （2）非订单软件（未签订合同）

计算机工程（Computer Engineering）由硬件工程（Hardware Engineering）和软件工程（Software Engineering）组成。硬件工程是研究硬件生产和硬件管理的工程学科，其内容包括计算机及网络硬件的分析、设计、生产、采购、验收、安装、培训、维护。软件工程是研究软件生产和软件管理的工程学科，其内容包括市场调研、正式立项、需求分析、项目策划、概要设计、详细设计、编程、测试、试运行、产品发布、用户培训、产品复制、实施、系统维护、版本升级。由于软件的生产和管理比硬件复杂，积累的经验不如硬件那么丰富，所以软件工程的研究成为一个长期的热点。

【例 1-1】　请读者根据自身环境，规划、设计、安装一个校园网。这是一个硬件工程，其中要完成的工作内容包括：制订设计方案，网络设备的选型、配置、采购、验货、布线、安装、调试、运行和交付。在安装和调试中，要安装和调试许多软件，如网络操作系统、数据库管理系统、教学软件系统、办公自动化系统、防火墙及杀毒软件等。

由于有这么多的软件需要选型、配置、采购、安装、调试，所以在今天，除了生产硬件的厂商之外，纯粹的“硬件工程”几乎不存在，大多数硬件工程都与软件有关，于是就出现了一个新名词“网络工程”。它是介于硬件工程和软件工程之间的系统工程，人们有时也称它为“系统集成工程”。

2．软件定义

为了弄清软件工程的概念，首先要了解程序和软件的概念。一般认为，程序是计算机为完成特定任务而执行的指令的有序集合。站在应用的角度可以更通俗地理解为：

面向过程的程序 = 算法 + 数据结构

面向对象的程序 = 对象 + 消息

面向构件的程序 = 构件 + 构架

通常，软件有以下定义：

软件 = 程序 + 数据 + 文档

这里的“程序”，是对计算机任务的处理对象和处理规则的描述；这里的“文档”，是为了理解程序所需的详细描述性资料；这里的“数据”，主要是软件系统赖以运行的初始化数据。

上述定义看起来很简单，实际上却来之不易。表 1-3 列出了美国人对软件定义的认识过程。直到今天，仍然有少数人认为：“软件就等于程序”。这些人在软件开发过程中，上来就写程序，而不是写文档。软件工程大师 Roger S. Pressman 对这些人提出了尖锐的批评：“越早开始写代码的人，就是越迟完成代码的人。”

表 1-3　美国人对软件定义的认识过程

年　　代	对软件定义的认识
20 世纪 50 年代	软件就等于程序，软件系统就是程序系统
20 世纪 60 年代	软件等于程序加文档。这里的文档，是指软件开发过程中的分析、设计、实现、测试、维护文档，还不包括管理文档
20 世纪 70 年代	软件等于程序加文档再加数据，这里的数据不仅包括初始化数据、测试数据，而且包括研发数据、运行数据、维护数据，也包括软件企业积累的项目工程数据和项目管理数据中的大量决策原始记录数据

至于对管理文档的全面认识，那就更晚了。直到 1974 年，美国人才开始认识到软件需要管理。1984 年，美国人开始认识到软件管理是一个过程管理，或是一个管理过程。1991 年，出现了软件过程能力成熟度模型 CMM（Capability Maturity Model for Software）1.0 版，人们研究了软件过程管理的具体内容与方法，并将软件开发和管理中产生的各种文档称为“软件工作产品”，而将最后交付给用户使用的软件工作产品称为“软件产品”。1996 年，出现了统一建模语言 UML 0.9 版，称软件管理文档为“管理制品”，称软件开发文档为“技术制品”，两者合称为“制品（Artefact）”。

3．文档的重要性

文档在软件工程中特别重要，文档是否规范与齐全，是衡量软件企业是否成熟的重要标志之一。软件文档分为开发文档和管理文档两大类。开发文档主要由项目组书写，用于指导软件开发与维护；管理文档主要由软件工程管理部门书写，用于指导软件管理和决策。两类文档的标准、规范和编制模板，在全公司范围内要统一，这一工作由软件工程管理部门完成。开发文档是指导软件开发与维护的文档，开发与维护中所有的程序都是按照开发文档的要求编写与实现的。软件工程规定：文档必须指挥程序，而决不允许程序指挥文档；文档与程序必须保持高度一致，而决不允许程序脱离开文档。

开发文档本身具有严格的层次关系和依赖关系，这种关系反映在以下覆盖关系中：

（1）《目标程序》覆盖《源程序》。

（2）《源程序》覆盖《详细设计说明书》。

（3）《详细设计说明书》覆盖《概要设计说明书》。

（4）《概要设计说明书》覆盖《需求分析规格说明书》。

（5）《需求分析规格说明书》覆盖《用户需求报告》。

（6）《用户需求报告》覆盖《软件合同》/《软件任务书》。

管理文档本身具有严格的时序关系，这种时序关系反映在以下软件过程中，而过程由一系列的时间序列所组成：

（1）需求分析过程管理文档。

（2）软件策划过程管理文档。

（3）软件设计过程管理文档。

（4）软件实现过程管理文档。

（5）软件测试过程管理文档。

（6）软件维护过程管理文档。

（7）软件过程改进管理文档。

成熟的软件企业，都有一套自己的开发文档和管理文档编写标准或编写模板，在企业内部严格执行。

4．软件的最新定义

软件的最新定义如下：

软件 = 知识+程序+数据+文档

定义中增加了“知识”。这里的“知识”，主要指各种各样的相关行业领域的专业知识。实际上，知识只是网络的外在表现，程序、数据、文档才是网络的内在实质。也就是说，知识是通过程序、数据、文档来实现的。

对这一定义的另外一种解释是：软件到底是什么呢？软件就是网络，网络就是知识，知识就是信息。站在网民的角度看，软件就是知识加信息；站在程序员角度看，软件就是程序加数据；站在软件管理者角度看，软件就是数据加文档。

网络是知识的载体，知识是网络的灵魂。

1.2　软件工程的定义

1．软件危机

软件工程来源于软件危机，即先有软件危机，后有软件工程。

20 世纪 60 年代中期，在美国出现了软件危机（Software Crisis），它表现在研发大型软件时，软件开发的成本增大、进度延期、维护困难和质量得不到保障。最突出的例子是美国 IBM 公司于 1963—1966 年开发的 IBM360 系列机操作系统。该软件系统花了大约 5000 人年的工作量，最多时有 1000 人投入开发工作，源程序代码近 100 万行。尽管投入了这么多的人力和物力，得到的结果却极其糟糕。据统计，该操作系统每次发行的新版本，都是从前一个旧版本中找出 1000 个程序错误而修正的结果。可想而知，这样的软件质量糟糕到什么地步。

由此可见，所谓软件危机，就是在软件开发和维护过程中所遇到一系列难以控制的问题。“软件危机”这个专业术语是于 1968 年由 NATO（North Atlantic Treaty Organization，北约）的计算机科学家在德国召开的国际学术会议上首次提出的。

为了克服软件危机，同样是在 1968 年，北约科技委员会召集了近 50 名一流的编程人员、计算机科学家和工业界巨头，讨论和制定摆脱“软件危机”的对策。在这次会议上，第一次提出了软件工程（Software Engineering）这个专业术语。当时，人们的想法是：若借用建筑工程或机器制造工程的思想、标准、规范、规程去开发软件与维护软件，也许能克服软件危机。以后的实践证明：用工程的方法开发软件与维护软件是个好主意，但是要完全克服软件危机，还有许多其他工作要做。

2．软件工程定义

1993 年，权威杂志 IEEE 对软件工程的定义是：软件工程是将系统化的、严格约束的、可量化的方法，应用于软件开发、运行和维护中。

2001 年，软件工程大师 Roger S. Pressman 对软件工程的定义是：软件工程是一个过程、一组方法和一系列工具。

由于软件技术飞速发展，所以软件工程的定义也要与时俱进。下面根据当前软件技术的进展状况，给出现代软件工程的最新定义。

软件工程是研究软件开发和软件管理的一门工程学科。

这里，一是强调开发。开发是软件工程的主体，开发是在规定的时间、按照规定的成本、开发出符合规定质量要求的软件。二是强调管理或过程管理。当然，开发中有管理，管理是为了更好地开发。所以开发和管理是一个问题的相辅相成的两个方面。许多软件项目的失败，不是在开发技术上出了问题，而是在管理过程上出了问题。所以在某种程度上说，对于一个软件企业，过程管理比开发技术更重要。三是强调工程。要将软件的开发（包括维护）当成一项工程，既要按照工程的办法去开发，又要按照工程的办法去管理。四是强调学科。时至今日，软件工程不只是一门课程，而是一个学科体系，即软件工程知识体系。

3．软件工程学科体系

软件工程作为一个学科体系，到 21 世纪初才初步形成。2001 年 4 月 18 日，美国发布了软件工程知识体系指南 SWEBOK（guide to the Software Engineering Body of Knowledge）0.95 版。2004 年，软件工程学科体系的内容才基本确立，就在这一年，美国 ACM 和 IEEE-CS 联合制订了 SWEBOK 2004 版，它将软件工程学科体系的知识划分为如下 10 个知识域。

（1）软件需求（Software Requirements）。软件需求是真实世界中的问题而必须展示的特性。软件需求知识域有 7 个子域：需求基础、需求过程、需求获取、需求分析、需求规格说明、需求确认、实践考虑。

（2）软件设计（Software Design）。软件设计既是定义一个系统的体系结构、组件、接口和其他特征的过程，又是这个过程的结果。软件设计知识域有 6 个子域：软件设计基础、软件设计关键问题、软件结构与体系结构、软件设计质量的分析与评价、软件设计符号、软件设计的策略与方法。

（3）软件构造（Software Construction）。它指通过编码、验证、单元测试、集成测试和排

错的组合，具体创建一个可以工作的、有意义的软件。其知识域有3个子域：软件构造基础、管理构造、实际考虑。

（4）软件测试（Software Testing）。它由在有限测试用例集合上，根据期望的行为对程序的行为进行的动态验证组成，测试用例是从实际上无限的执行域中适当选择出来的。软件测试知识域有5个子域：软件测试基础和测试级别、测试技术、需求分析、与测试相关的度量、测试过程。

（5）软件维护（Software Maintenance）。软件一旦投入运行，就可能出现异常，运行环境可能发生改变，用户会提出新的需求。生命周期中的软件维护，从软件交付时开始。软件维护的知识域有4个子域：软件维护基础、软件维护的关键问题、维护过程、维护技术。

（6）软件配置管理（Software Configuration Management）。软件配置是为了系统地控制配置的变更，维护软件在整个系统生命周期中的完整性及可追踪性，而标志软件在不同时间点上的配置的学科。软件配置管理知识域有6个子域：软件配置管理过程管理、软件配置标志、软件配置控制、软件配置状态统计、软件配置审核、软件发行管理和交付。

（7）软件工程管理（Software Engineering Management）。进行软件工程的管理与度量，虽然度量是所有知识域的一个重要方面，但是这里所说的是度量程序的主题。软件工程管理知识域有6个子域：启动和范围定义、软件项目计划、软件项目实施、评审与评价、关闭、软件工程度量。前5个覆盖软件过程工程管理，第6个描述软件度量的程序。

（8）软件工程过程（Software Engineering Process）。涉及软件工程过程本身的定义、实现、评定、度量、管理、变更和改进。软件工程过程知识域有4个子域：过程实施与改变、过程定义、过程评定、过程和产品度量。

（9）软件工程工具和方法（Software Engineering Tool and Method）。它有软件工程工具、软件工程方法两个子域。

（10）软件质量（Software Quality）。处理跨越整个软件生命周期过程的软件质量的考虑，由于软件质量问题在软件工程中无处不在，其他知识域也涉及质量问题。软件质量知识域有3个子域：软件质量基础、软件质量过程、实践考虑。

在上述软件工程学科体系中，前5个知识域是讲软件开发，后5个知识域是讲软件管理。由此可见，软件工程知识体系包括软件开发和软件管理两大部分，所以软件工程的定义也应该包括软件开发和软件管理两项内容。

4．软件工程课程研究的内容

软件工程课程与软件工程学科体系是有区别的：前者是一门或一组课程，后者是一个知识体系；前者是一个局部问题，后者是一个整体问题。

作为一门软件工程课程，它研究的内容至今没有统一的说法。可以这么认为，软件工程课程研究的内容应该涵盖“软件生命周期模型、软件开发方法、软件支持过程、软件管理过程、软件工程标准与规范”这5个方面，如表1-4所示。

尽管软件生命周期模型和软件支持过程非常重要，但是现代软件工程研究的重点，仍然是软件开发方法和软件管理过程。在软件管理过程的内容中，除了ISO 9001和CMMI之外，还将软件企业文化也列入其中，如微软企业文化、敏捷文化现象和IBM企业文化。

软件工程标准是对软件产品的约束，例如软件产品的界面标准、包装标准、文档标准、

测试标准、评审标准、鉴定标准等。软件工程规范是对软件开发人员行为的约束，例如命名规范、需求规范、设计规范、编码规范、维护规范等。在软件企业内部，企业管理人员特别重视软件工程的标准与规范。为此，每个大型的软件企业，根据自身的特点，都制定并发布了自己的软件工程标准与规范，在自己企业内部严格执行。

表 1-4　软件工程课程研究的内容

序号	研究方面	具体内容
1	软件生命周期模型	如：瀑布模型、增量模型、原型模型、迭代模型、XP 模型
2	软件开发方法	如：面向过程的方法、面向元数据的方法、面向对象的方法
3	软件支持过程	如：CASE 工具 Rose、北大青鸟系统、Power Designer、ERWin
4	软件管理过程	如：CMMI、软件企业文化、敏捷（XP）文化现象
5	软件工程标准与规范	如：命名标准与规范、设计标准与规范、编程标准与规范

【例 1-2】　请读者开发一个“图书馆信息系统”，即图书馆 MIS。这是一项小型软件工程，为了完成这项任务，读者首先要选择软件生命周期模型，确定开发方法，准备开发工具，设计开发环境和运行环境；然后进行需求分析、概要设计、详细设计、编程、测试、试运行、正式运行、验收和交付；最后是系统维护或系统升级换代。这样，读者就按照所选择的开发模型，走完了软件的一个生命周期。这一系列的软件开发过程和管理过程，就是软件工程。

5．软件工程基本原理

习惯上，人们常常把软件工程的方法（开发方法）、工具（支持方法的工具）、过程（管理过程）称为软件工程三要素，而把美国著名的软件工程专家 B.W. Boehm 于 1983 年提出的 7 条原理作为软件工程的基本原理。

（1）用分阶段的生命周期计划严格管理软件开发。阶段划分为计划、分析、设计、编程、测试和运行维护。

（2）坚持进行阶段评审。若上一阶段评审不通过，则不能进入下一阶段开发。

（3）实行严格的产品版本控制。

（4）采用现代程序设计技术。

（5）结果应能清楚地审查。因此，对文档要有严格要求。

（6）开发小组的成员要少而精。

（7）要不断地改进软件工程实践的经验和技术，要与时俱进。

上述 7 条原理，虽然是在面向过程的程序设计时代（结构化时代）提出来的，但是，直到今天，在面向元数据和面向对象的程序设计新时代，它仍然有效。根据“与时俱进”的原则，还有一条基本原理在软件的开发和管理中特别重要，需要补充进去，作为软件工程的第 8 条基本原理。

（8）二八定律。

在软件工程中，所谓二八定律，就是一般人常常将 20%的东西误以为是 80%的东西，而将 80%的东西误以为是 20%的东西。

例如，对软件项目进度和工作量的估计：一般人主观上认为已经完成了 80%，但实际上只完成了 20%；对程序中存在问题的估计：一般人不知道 80%的问题存在于 20%的程序之中；对模块功能的估计：一般人不知道 20%的模块，实现了 80%的功能；对人力资源的估计：一

般人不知道20%的人，解决了软件中80%的问题；对投入资金的估计：一般人不知道信息系统中80%的问题，可以用20%的资金来解决。

在软件开发和管理的历史上，有无数的案例都验证了二八定律。所以，软件工程发展到今天，作者认为，它的基本原理共有8条。

在软件工程中，有些定理或定律是不需要从理论上严格证明的，只需要在实践中检验。因为实践是检验真理的唯一标准，从来没有人去证明过二八定律，但它却是放之四海而皆准的真理。

研究二八定律的现实意义是，指导软件开发计划的制订与执行。如果事先掌握了二八定律，就能自觉地用二八定律去制订、跟踪与执行软件开发计划。也就是说，计划中要用开始的20%时间，去完成80%的开发进度；剩下20%的进度，要留下80%的时间去完成。只有这样，项目的开发计划与项目的开发进度才能吻合。这是为什么呢？就是因为软件中的许多问题，只有到软件开发后期才会真正暴露出来！这就是软件的特点，这就是软件开发工作的规律！

6．软件工程在中国

软件工程是何时来到中国大陆的？国内实时操作系统的奠基人孟庆余教授在《银河精神》的回忆录中记述（2003年9月28日）如下：

1982年，软件工程的创始人、美籍华人叶祖尧博士，带着自己开创的“软件工程学”理论来到中国，成为当时中国政府计算机领导小组的顾问。他制定了一项“中国软件发展计划”，提交给当时的国务院主要领导。从此，软件工程在中国开始启动，并且一发而不可收。

1984年，国家科委在北京召开“软件工程”大会。会议期间，国防科技大学陈火旺院士与孟庆余教授，宴请了美国软件工程专家叶祖尧博士。席间，时任美国马里兰大学计算机系主任的叶祖尧博士说：“软件工程，只有你们长沙（国防科技大学）并行机的研究搞得最好！”

20世纪80年代银河系列巨型机的发展历史已经证明，21世纪天河系列巨型机的发展历史再次证明：叶祖尧博士的话是对的。

另外，根据《软件工程技术概论》（北京：科学出版社，2002）一书的记载，中国软件工程的第一本书籍，是由朱三元等人编著的《软件工程指南》，出版时间为1985年。

1.3　软件工程在软件行业中的作用

软件工程是软件行业的一门工程管理科学，更是系统分析员和项目经理以上人员必备的知识体系，为了将我国的软件产业搞上去，使软件产业成为国民经济的支柱产业，使中国早日成为一个软件大国与软件强国，在软件界怎么强调软件工程也不过分。

【例1-3】　20世纪90年代初，有两个软件团队，一个较大（10多人），另一个较小（6人），都在开发财务系统。

较大的那个团队，工作不规范，没有文档，没有评审，也没有团队协作精神，结果开发出来的产品可维护性差，没有打开市场，没有产生经济效益和社会效益，致使产品与团队最后同归于尽。

较小的那个团队，同舟共济，工作规范，有正规文档，有阶段评审，分工明确：一人负

责原始凭证和输出报表的收集、归类和整理，这实际上是做需求分析；一人负责科目和数据字典（代码表），这实际上是做信息的标准化与规范化；一人负责记账凭证的录入和修改，这实际上是做数据库的设计和加载工作；一人负责日记账、明细账和总账之间的平衡与对账，这实际上是做数据处理；一人负责统计、报表和查询，这实际上是做数据输出工作；一人负责总体设计和项目管理，这就是项目经理的工作。他们的工作进度虽然不快，但最后形成了产品，打开了市场，产生了经济效益和社会效益，并且发展成为一个大型 IT 企业，这 6 个人后来都成了业界精英。

造成这两个团队不同结果的原因是什么呢？一个根本原因，就是较大的团队没有软件工程知识和团队协作精神，较小的团队有一些软件工程知识和很强的团队协作精神。由此可见，软件工程知识背景和团队精神是多么重要。实际上，团队精神是一种软件企业文化，软件企业文化属于软件过程管理的范畴，软件过程管理是软件工程研究的四大内容之一。

因为软件工程来自于软件行业，又服务于软件行业，所以下面主要讨论它在软件行业中的作用。

从历史上讲，软件工程的作用，是为了克服 20 世纪 60 年代出现的软件危机。

从当前来讲，软件工程的作用，就是告诉人们怎样去开发软件和管理软件。具体地讲，它表现在与软件开发和管理有关的人员和过程上。为了说明这个问题，首先，分析软件行业的人才结构，看看这些人员的工作与软件工程有什么关系。

一般来说，软件行业的专业人才由下列几个层次组成。

（1）高层管理人员。他们应具备的基本条件是：软件专业宏观知识、软件工程管理知识，加上商业与资本运作知识。他们要用软件工程的理论和方法，来管理整个公司的软件业务。

（2）中层项目经理和软件工程师。他们应具备的基本条件是：系统分析知识、系统设计知识，加上项目管理知识。他们要用软件工程的理论和方法，来管理项目组的软件开发。他们的个人奋斗目标是软件管理专家、分析设计专家、开发技术专家，他们是软件工程的实践者。

（3）软件蓝领工人。他们应具备的基本条件是：掌握阅读文档的技能、程序设计的技巧，加上软件测试的知识。他们要用软件工程的理论和方法，来实现软件项目的软件功能、性能、接口、界面。

（4）软件营销人员。他们应具备的基本条件是：营销知识、售前知识，加上软件工程基本知识。他们要用软件工程的基本思路，来与客户进行沟通，以赢得客户的信任。

（5）软件实施和维护人员。他们应具备的基本条件是：软件客户化及安装、运行、维修技术。他们要用软件工程的基本方法，来实现软件功能、性能与接口的实施和维护。

（6）软件售前人员。他们是软件公司的产品形象代表，其奋斗目标是：既要成为某个行业领域的产品专家，又要成为该产品的实现顾问。只有这样，他们才能看懂招标书、写好投标书、讲好投标书。在制作和宣讲投标书的过程中，有许多与软件工程相关的知识和内容，如项目开发方法、开发工具、开发环境、运行环境、管理方法、质量和进度控制方法，只有把这些方法写清讲透，用户才能相信认可，投标才有成功把握。这些知识和内容，离不开软件工程知识的学习。

在以上 6 种人员中，软件工程这门课是前三种人员的必修课。对后三种人员，若想在工作中寻求更大的发展空间，则应提升自己的知识结构和工作层次，且十分需要掌握软件工程的基本知识。当然，对于不同岗位，知识结构要求有所不同，侧重点也不同。但是，只要在软件

行业工作，就会自觉或不自觉地参与软件岗位竞争，就必须重视软件工程，学好软件工程，用好软件工程，不断地将自己的实践经验上升到软件工程的理论与方法，又不断地用软件工程的理论与方法指导自己的实践活动，使自己不断地得到升华和发展，这就是软件工程的作用。

从软件项目团队来讲，软件工程的作用是：在规定的时间内，按照规定的成本，完成预期质量目标（软件的功能、性能和接口达到需求报告标准）的软件。

从软件企业本身来讲，软件工程的作用是：持续地规范软件开发过程和软件管理过程，不断地优化软件组织的个人素质和集体素质，从而逐渐增强软件企业的市场竞争实力。

从软件大国与强国来讲，软件工程的作用是：它在一个国家的计算机界及软件界的普及与推广，可以使这个国家变为一个软件大国，进而变为一个软件强国。

由于软件工程的作用越来越大，它的地位也越来越高。以前，软件工程在高校只是一门课程。现在，它作为一个学科体系，设立了软件工程专业和软件工程学士、硕士、博士学位。

1.4 软件工程方法论

1.4.1 软件工程方法论的提出

长期以来，人们将软件开发方法与软件生命周期模型，甚至将软件开发方法与软件过程改进模型混为一谈，因而误认为软件生命周期模型或软件过程改进模型就是软件开发方法。例如，将迭代模型 RUP（Rational Unified Process）和过程改善模型 CMMI（Capability Maturity Model Integration）误认为软件开发方法或软件工程方法论。事实上，软件开发方法与软件生命周期模型是不同的，软件开发方法与软件过程改进模型就更不相同了。软件开发方法学源于程序设计语言方法学，而软件生命周期模型或软件过程改进模型与程序设计语言方法学无关。

软件生命周期模型是指在整个软件生命周期中，软件开发过程应遵循的开发路线图。或者说，软件生命周期模型是软件开发全部过程、活动和任务的结构框架。

例如，瀑布模型、增量模型、螺旋模型、喷泉模型、XP 模型、原型模型和 RUP 迭代模型，它们都有各自清晰的开发路线图，规定了各自的开发过程、活动和任务的结构框架。关于这些软件生命周期模型，将在后续的有关章节中详细论述。

软件开发方法是指在软件开发路线图中，开发人员对软件需求、设计、实现、维护所采用的开发思想、开发技术、描述方法、支持工具等。

在软件工程方法学方面，大体可分为程序设计方法学和软件开发方法学，前者是关于小规模程序的设计方法学，后者是关于大规模软件的开发方法学。

例如，在程序设计方法学中，最基本的方法有面向过程程序设计方法、面向对象程序设计方法、面向元数据（Meta-data）程序设计方法。在软件开发方法学中，最基本的方法有面向过程方法、面向对象方法、面向元数据方法、形式化方法。它们都是软件开发方法，都有各自的开发思想、开发技术、描述方法、支持工具等。

软件工程中软件开发方法的集合，称为**软件工程方法论**。

现在的问题是：到目前为止，在软件工程方法论中，到底包括哪几种最基本的软件开发方法？这几种开发方法到底存在什么关系？下面将回答这些问题。

1.4.2　面向过程方法

面向过程的方法（Procedure-Oriented Method），来自于面向过程的程序设计语言，如汇编语言、C 语言。面向过程方法包括面向过程的需求分析、设计、编程、测试、维护、管理等。

面向过程方法，习惯上称为传统软件工程开发方法或结构化方法。它包括结构化分析、结构化设计、结构化编程、结构化测试、结构化维护。面向过程方法，有时又称面向功能的方法，即面向功能分析、设计、编程、测试、维护。由此可见，面向过程方法、面向功能方法、结构化方法，三者是同一个意思。

在软件工程发展史上，曾经出现过的面向过程方法有：

（1）面向结构化数据系统的开发方法 DSSD（Data Structured Systems Development）。

（2）面向可维护性和可靠性设计的 Parnas 方法。

（3）面向数据结构设计的 Jackson 方法。

（4）面向问题设计的 PAM 方法。

（5）面向数据流方法。

上述 5 种方法的详细内容，利用百度或 Google 搜索引擎，都可以在网上查到。但是，不管方法名是什么，这 5 种方法在宏观上都属于面向过程方法，支持这些方法的是面向过程的结构化编程语言。

面向过程方法设计中强调模块化思想，采用“自顶向下，逐步求精”的技术对系统进行划分，分解和抽象是它的两个基本手段。面向过程方法编程时采用单入口单出口的控制结构，并且只包含顺序、选择和循环三种结构，目标之一是使程序的控制流程线性化，即程序的动态执行顺序符合静态书写结构。

在面向过程的 5 种具体方法中，面向数据流方法最具有代表性。

面向数据流的设计方法，把数据流图映射成为软件结构图，数据流图的类型决定了映射方法，数据流图 DFD（Data Flow Diagram）可以分为变换型数据流图和事务型数据流图。

（1）具有明显的输入、变换（加工）和输出界面的数据流图称为变换型数据流图。

（2）数据沿输入通路到达一个处理模块，这个处理模块根据输入数据的类型在若干动作序列中选出一个来执行，这类数据流图称为事务型数据流图，并且称这个模块为事务中心。它完成如下任务：接收输入数据；分析数据并确定数据类型；根据数据类型选取一条活动通路。

关于面向数据流的方法，在本书的后续章节中还会有进一步的介绍。

软件的基础是程序，没有程序就没有软件，也就没有软件工程方法。对于软件行业来说，某一种软件工程方法往往来自于某一类程序设计语言。面向过程的方法，来自于 20 世纪 60～70 年代流行的面向过程的程序设计语言，如 ALGOL、PASCAL、BASIC、FORTRAN、COBOL、C 语言等，这些语言的特点是：用“顺序、选择（if-then-else）、循环（do-while 或 do-until）”三种基本结构来组织程序编制，实现设计目标。面向过程方法开始于 20 世纪 60 年代，成熟于 70 年代，盛行于 80 年代。该方法在国内曾经十分流行，大量应用，非常普及。

面向过程方法的优点是：以处理流程为基础，简单实用。其缺点是：只注重过程化信息，忽略了信息的层面关系及相互联系。它企图使用简单的时序过程方法（顺序、分支、循环三种结构），来描述关系复杂（随机）的信息世界，因而对于关系复杂的信息系统来说，其描述能力不强，最后可能导致软件设计、开发和维护陷入困难。

自从面向对象方法出现之后，在许多领域，面向过程方法逐渐被表述能力更强的面向对象方法所取代。当前，面向过程方法主要用在过程式的程序设计中，如对象方法（函数）、科学计算、实时跟踪和实时控制的实现。

【例 1-4】 面向过程方法在军事领域的实时跟踪监控系统中有很好的应用。如我方侦察卫星发射后其飞行轨迹的捕获、测量、跟踪和预报，导弹防御系统中敌方导弹发射后飞行轨迹的捕获、测量、跟踪和预报，其软件系统都是采用面向过程方法设计和实现的。使用面向过程方法，系统的执行路径可由系统自动控制，也就是程序自动控制，这是一切自动控制与跟踪系统所必需的。

1.4.3　面向对象方法

面向对象方法（Object-Oriented Method），在不少教材中称为现代软件工程开发方法。该方法包括面向对象的需求分析、设计、编程、测试、维护、管理等。面向对象方法是一种运用对象、类、消息传递、继承、封装、聚合、多态性等概念来构造软件系统的软件开发方法。

面向对象方法的特点是，将现实世界的事物（问题域）直接映射到对象。分析设计时由对象抽象出类（Class），程序运行时由类还原到对象（Object）。

面向对象方法，源于 20 世纪 80 代年开始流行的面向对象的程序设计语言，如 Java、C++、Delphi、Visual Basic 语言等。面向对象方法的基本特点是，将对象的属性和方法（即数据和操作）封装起来，形成信息系统的基本执行单位，再利用对象的继承特征，由基本执行单位派生出其他执行单位，从而产生许多新的对象。众多的离散对象通过事件或消息连接起来，就形成了软件系统。20 世纪 80 年代末，微软视窗操作系统的出现，使它产生了爆炸性的效果，大大加速了它的发展进程。90 年代中期，UML（Unified Modeling Language）和 Rose（Rational object-oriented system engineering）的产生，标志着它逐步走向了成熟，并且开始普及。21 世纪初，面向对象的两类开发平台是.Net 平台和 J2EE 平台。

面向对象方法，实质上是面向功能方法在新形势下（由功能重用发展到代码重用）的回归与再现，是在一种高层次（代码级）上新的面向功能的方法论，它设计的“基本功能对象（类或构件）”，不仅包括属性（数据），而且包括与属性有关的功能（或方法，如增加、修改、移动、放大、缩小、删除、选择、计算、查找、排序、打开、关闭、存盘、显示和打印等）。它不但将属性与功能融为一体，而且对象之间可以继承、派生以及通信。因此，面向对象设计，是一种新的、复杂的、动态的、高层次的面向功能设计。它的基本单元是对象，对象封装了与其有关的数据结构及相应层的处理方法，从而实现了由问题空间到解析空间的映射。一句话，面向对象方法也是从功能入手，将功能与方法作为分析、设计、实现的出发点和最终归宿。

面向对象方法的优点是：能描述无穷的信息世界，同时易于维护。其缺点是：对于习惯于面向过程方法的人，较难掌握。

面向对象方法是当前软件界关注的重点，是软件工程方法论的主流。面向对象的概念和应用已超越了程序设计和软件开发，扩展到更宽的范围，如交互式界面、应用结构、应用平台、分布式系统、网络管理结构、CAD 技术、人工智能等领域。

业界流传的面向方面方法、面向主体方法和面向架构方法，都是面向对象方法的具体应用实例。

【例 1-5】　互联网上各种网站的设计、实现和维护，都是面向对象方法的案例。游戏软件的设计、实现和维护，更是面向对象方法的杰作。

【例 1-6】　面向对象方法在电子商务中的应用有：网站前台界面的制作，信息的发布和处理，用户在网上浏览和录入信息等应用软件都是利用面向对象的方法设计与实现的。个人网页的制作也是面向对象方法的应用例子。窗口操作系统与互联网的出现，为面向对象方法开辟了无限的前景。

1.4.4　面向元数据方法

这里讲的面向元数据方法（Meta-data Oriented Method），既不是传统软件工程中的“面向数据流”方法，也不是传统意义上的面向数据结构的 Jackson 方法，它们都是面向过程方法，而且这两种方法都出现在关系数据库管理系统 RDBMS（Relational Database Management System）成熟之前。所以这里讲的面向元数据方法，就是面向元数据方法，它与面向过程方法截然不同。

面向元数据方法来源于面向元数据的程序设计思想，即关系数据库语言的程序设计思想。当关系数据库管理系统和数据库服务器出现之后，面向元数据方法才被人们所发现与重视。当数据库设计的 CASE 工具 Power Designer、Oracle Designer 和 ERWin 出现之后，面向元数据设计方法才开始流行。因为计算机就是网络，网络就是服务器，多数服务器都是数据库服务器，数据库服务器上的软件开发就采用面向元数据方法，由此可见面向元数据方法的重要性。

元数据（Meta-Data）是关于数据的数据、组织数据的数据、管理数据的数据。

这里的元数据，泛指一切组织数据的数据，如类的名称、属性和方法，实体的名称、属性和关联，数据库中的表名、字段名、主键、外键、索引、视图，数据结构中存储数据的框架等。但是，我们研究的重点，是指数据库中的元数据。

面向元数据方法，就是在软件需求分析、设计、实现、测试、维护过程中，均以元数据为中心的软件工程方法。

例如，数据库概念设计中的实体名和属性名，数据库物理设计中的表名和字段名，它们就是元数据。而具体的某一个特定的实例，它们不是元数据，而是对象或记录，是被元数据组织的数据。关系数据库管理系统自带的程序设计语言，提供了强大的面向关系表中数据的编程能力，典型的例子就是编写存储过程（Stored Procedure）和触发器（Trigger）。Oracle 数据库管理系统自带的编程工具 Developer 2000，是一个面向元数据的编程工具。Oracle Designer 加上 Developer 2000，构成了一个完整的面向元数据的信息系统开发环境。

面向元数据方法包括面向元数据的需求分析、设计、编程、测试、维护。

（1）面向元数据的需求分析，是在需求分析时，找出信息系统所有的元数据，使其完全满足信息系统对数据存储、处理、查询、传输、输出的要求。也就是说，有了这些元数据，信息系统中的一切原始数据不但都被组织起来，而且能完全派生出系统中的一切输出数据。

（2）面向元数据设计，是利用需求分析获得的元数据，采用面向元数据的 CASE 工具，设计出信息系统的概念数据模型 CDM（Conceptual Data Model）和物理数据模型 PDM（Physics Data Model），以及从原始数据到输出数据的所有算法与视图。

（3）面向元数据编程，是在物理数据模型 PDM 的基础上，根据信息系统的功能、性能、接口和业务规则，建立数据库表和视图，再利用数据库编程语言，编写出存储过程和触发器。

（4）面向元数据测试，是对数据库表初始化并加载之后，运行相关的存储过程和触发器，测试信息系统的各种功能需求与性能指标。

（5）面向元数据维护，是对数据库表中的记录进行统计、分析、审计、复制、备份、恢复，甚至对表结构及视图结构进行必要的调整。

事实上，20 多年来，面向元数据方法已经是建设信息系统、数据库、数据仓库和业务基础平台的基本方法。概括起来，面向元数据方法的要点是：

（1）数据（Data）位于企业信息系统的中心。信息系统就是对数据的输入、处理、传输、查询和输出。

（2）只要企业的业务方向和内容不变，企业的元数据就是稳定的，由元数据构成的数据模型（Data Model）也是稳定的。

（3）对元数据的处理方法是可变的。用不变的元数据支持可变的处理方法，即以不变应万变，这是企业信息系统工程的基本原理。

（4）信息系统的核心是数据模型。数据模型包括概念数据模型 CDM 和物理数据模型 PDM。数据模型的表示形式是 E-R 图，它用 CASE 工具设计。例如，Power Designer，Oracle Designer 或 ERWin，它们不但具有正向设计功能，而且具有逆向分析功能，这样才能实现快速原型法。

（5）信息系统的编程方法主要是面向对象（除数据库服务器层面上），其次才是面向元数据（在数据库服务器层面上）和面向过程（在实现存储过程和对象方法中）。

（6）用户自始至终参与信息系统的分析、设计、实现与维护。

面向元数据方法的优点是：通俗易懂，特别适合信息系统中数据层（数据库服务器）上的设计与实现。其缺点是：只能实现二维表格，不能实现窗口界面。

面向元数据方法，是作者在 IT 企业多年软件工程管理经验与在高校多年软件工程教学经验的积累、反思与升华。该方法与关系数据库管理系统紧密地捆绑在一起，只要面向对象数据库不能完全替代关系数据库，这种方法就不会终结。目前，数据库管理系统的发展趋势是：在关系型数据库的基础上，将面向对象的某些特性（如继承）添加上去，称为“对象-关系型数据库”，但本质上仍然是一个关系型数据库。

【例 1-7】 面向元数据方法在电子商务中也有应用。网站后台数据库服务器上的数据处理和数据传输，其软件都是利用面向元数据方法设计与实现的。实际上，不管网络应用系统是 C/S 结构还是 B/S 结构，在数据库服务器（S）上对数据的分析、设计和实现，都自觉或不自觉地使用了面向元数据方法。

*1.4.5　形式化方法

1．什么是形式化方法？

软件工程的形式化方法（Formalized Method），是建立在严格的数学基础上、以逻辑推理为出发点、具有精确数学语义的开发方法。

软件工程中的形式化方法是软件工程的研究领域之一，其内容包括：有限状态机、State charts、Petri 网、通信顺序进程、通信系统演算、一阶逻辑、程序正确性证明、净室软件工程、时态逻辑、模型检验、Z 形式规约语言、B 语言和方法、VDM 系统、Larch 等。

作为一种以数学逻辑为基础的方法，形式化方法以其严密性越来越受到众多领域的重视，尤其是在安全

性和可靠性作为关键问题的系统，如核电站、航空航天、铁路运输系统中得到了较为广泛的应用。但是，对于形式化方法在工业领域的实际应用问题，在软件工程界，尤其是在系统开发人员中，还存在着相当多的疑问。

1990 年，J.A. Hall 回答了有关形式化方法的以下 7 个疑问。

（1）该方法可否保证软件系统的完美无缺？答：形式化方法不能保证系统的完美无缺，也并不能减少系统所需的测试。用户不能认为它是万能的。

（2）它处理的只是程序正确性的证明？答：形式化方法不仅仅局限于对程序正确性的证明。

（3）它只适用于“安全第一”的系统？答：形式化方法不仅仅局限于“安全第一”的系统。

（4）它需要专业的数学知识？答：许多复杂问题的简单形式化描述，以及若干项目的成功运作反驳了有关形式化方法需要专业的数学知识。

（5）它增加系统开发的成本？答：是的，增加了研发成本。

（6）用户无法接受它？答：最终用户以及非专业人员，在系统开发中的广泛参与，说明了用户对该方法的认可。

（7）无法应用于大型的实际系统？答：它在几个大型实际系统中成功应用，已经引起了广泛的关注，也否定了形式化方法无法应用于大型实际系统的说法。

1995 年，Jonathan P. Bowen 进一步回答了随着计算科学的发展，有关形式化方法的以下其他 7 个新疑问。

（8）该方法延迟开发进程？答：尽管一些应用形式化方法的项目由于各种原因被延迟，但是多数都因该方法的成功应用显著地缩短了开发时间。

（9）它缺乏支持工具？答：随着形式化方法领域的不断壮大，对其支持的工具会越来越丰富。类似于 CASE 的集成工具也已经出现。

（10）它将代替传统的工程设计方法？答：它与结构化软件工程方法并不是相互对立，相反，二者的结合将会是有益的相互补充。

（11）只适用于软件设计？答：该方法不仅适用于软件开发，同样适用于硬件设计，而且它使软、硬件联合设计成为可能。

（12）实际上并不需要它？答：尽管关于该方法的必要性有很多争论，但不可否认的是，在一些领域中，它是必需的，而且这些领域将越来越广泛。

（13）它缺乏支持？答：它正在为越来越多的人所接受与支持。在一些国家，该方法正逐渐步入大学的课堂。

（14）该方法的热衷人员只使用形式化方法？答：必须承认，该方法并不是万能的。在一些特定领域，它并不适宜，用户界面设计就是一例。

当然，上述回答是站在正方的立场上的。若站在反方的立场上，至少存在这样一个事实：在当今社会上，很难找到几个 IT 企业，他们在软件需求、设计、实现、验证中，系统地采用了形式化方法。

2．形式化方法的优势

随着软件系统的功能越来越多，规模越来越大，其开发的复杂度也越来越高。这样，软件中出现错误的概率也随之增加，由于这些错误可能会带来时间、财产甚至是人的生命的损失，因此软件工程的一个主要目标就是希望软件的可靠性不会随着系统的复杂性增大而降低。与其他传统的软件开发方法相比，以数学为基础的形式化方法有着无法比拟的严密性，它能够帮助发现其他方法不容易发现的系统描述的不一致性、二义性或不完整性，有助于增加软件开发人员对系统的理解。

形式化方法在软件开发中能够起到的作用是多方面的。首先是对软件需求和设计的描述。系统分析人员将软件需求和设计用形式化语言，而不是自然语言描述出来，这样做主要有两方面的好处：一是对于开发人员而言，由于自然语言本身的局限，传统的需求与设计描述是不规范的，可能存在歧义，容易被错误理解，一旦在这个环节出错，那么编写的代码也就不可避免地会出现错误；二是对于系统分析人员而言，在用形式化语言描述需求设计规范之后，可以利用模型检查、定理证明等方法，来判断软件的设计是否一致、是否完整，并且预测系统是否会表现出预期的特点、做出正确的行为。这可以在早期发现软件开发过程中的错误，并获得及时改正。对于软件开发来说，错误发现得越晚，修改成本就越高，越是大型复杂的系统，越是如此。形式化方法虽然在开发早期可能成本较高，但是却可以大大降低后期的维护和验证的费用。对于编程而言，还可以考虑自动代码生成。对于一些简单的系统，形式化的描述有可能直接转换成可执行程序，这就简化了软件开发过程，节约了资源，减少了出错的可能性。形式化方法还可以用于程序验证，以保证程序的正确性。另外，对于测试来讲，形式化方法可用于测试用例的自动生成，这既节省时间，又在一定程度上保证测试用例的覆盖率。

下面通过一个简单例子来看如何使用形式化的规格说明来对系统建模，并加以分析和验证。

现考虑以下需求：一个装冷却水的水箱在低水位传感器发出警告时需要向其中加水。每次加水都向水箱加 9 个单位的水。注意：

- 水箱最多能装 10 个单位的水。
- 每次读取水位的时间间隔会用掉 1 个单位的水。
- 低水位传感器会在水箱剩下 1 个单位或不到 1 个单位的水时发出警报。

以上描述涉及两个关键概念：水箱中的水位和水的使用量。我们可以将其形式化地描述如下：

（1）水位用整数表示，并且取值范围为[0,10]。

（2）水的使用量用整型常数 1 表示。

水位描述的是在任一时间水箱内的水量，而水的使用量是每次间隔使用的水量。

基本的需求是，如果水位为 1 个单位或者更少则向水箱加水，可以描述为：

（3）函数 fill 的输入/输出均为水位。如果输入为 L 个单位的水，fill 将在 L 等于或小于 1 时返回 L+9，其余情况返回 L。（这里定义了 fill(L)函数，用于说明水箱中的加水动作。）

根据常识，在这个系统中，一个时间间隔后新的水位应该是当前水位加上加入的水量减去这个时间间隔用掉的水量。设当前水位为 L，则：

（4）level = L + fill(L) – usage

我们检查这个规格描述是否与水位 level 的定义一致，也就是说，是否保证水位是 0～10 的一个整数。为了检查在第（3）条说明中定义的 fill 是否满足一致性，需要检查以下两条：

（5）对于所有的水位 L，

(L <= 1) 则 (0 <= L + 9) 并且 (L + 9 <= 10)

（6）对于所有的水位 L，

(0 <= L + fill(L) – usage) 并且 (L + fill(L) – usage <= 10)

这两个条件可以用形式化方法工具自动推理生成。其中第（5）条可以如下证明：

（5.1）因为 L >= 0，所以 L+9 >= 0。

（5.2）因为 L <= 1，所以 L+9 <= 10。

但是第（6）个条件不能保证成立，当 L = 9 时，

L + fill(L) = L + L – 1 = 9 + 9 – 1 = 17（不满足<= 10）

因此，以上的某一步出现了错误。仔细检查可以发现，第（4）条中关于一个时间间隔后的水位描述有问题：（4）level = L + fill(L) – usage（错误）。

这个描述与 fill 的定义不一致，因为 fill 返回的是水箱的新水位而不仅仅是加入的水量，因此我们将第（4）条改为

（4’）level = fill(L) – usage　（正确）

那么第（6）条判断就改为

（6’）对于所有的水位 L，

(0 <= fill(L) – usage) 并且 (fill(L) – usage <= 10)

这条判断留给读者自行验证。为了读者阅读方便，例中用自然语言而不是形式化语言来描述，读者可以通过相关参考书进一步研究如何用形式化语言来描述这个系统。

通过这个例子，读者可以看到，如何给出形式化的规格说明，如何检查一致性，如何定义系统行为，如何进行证明。

形式化方法本身是一个很灵活的方法，决策者不需要决定是否在系统中完全采用或者完全不采用这类方法，而可以根据自己的实际情况选择在开发的某些阶段，或者软件的某些模块某些功能点采用这一方法，也可以考虑将形式化方法与其他非形式化方法，如面向对象方法结合起来使用。

3．形式化方法的局限性

首先，形式化方法可以应用于软件开发的一些领域和阶段，但是它也有自己的适用范围。它基于离散数学，最适合对离散系统建模，尤其适用于包括很多逻辑交互的系统。如果一个系统包括很多不同的状态，状态之间的转移根据布尔条件来确定，这样的系统采用形式化方法收效明显；而如果系统结构比较松散，没有太多内在的逻辑关系，就算是形式化了，也得不到太大的好处。同时，形式化方法很难应用到采用数值化算法、需要大量计算的问题中，尤其是需要浮点运算的系统。

其次，目前，形式化方法只能解决中小规模的问题，如果系统规模较大，形式化方法的工作量会很大，并且有一定的难度。逻辑推理在形式化方法中占有很重要的位置，也是一个难点。逻辑推理主要包括模型检测和定理证明。模型检测是将系统建成一个有穷模型，然后检测该模型是否具备希望的性质。简单来说，就是在模型的状态空间中进行穷举搜索，主要的困难是，当系统非常庞大时采用何种数据结构和算法才能有效地进行搜索。

再次，形式化方法虽然比其他方法更加严密准确，能够通过数学推理对系统的特性做出证明，但是必须清醒地看到，即使一个系统完全采用了形式化方法，也不能说它就是百分之百正确的。因为在将非形式化的信息翻译成形式化表示的过程中，以及将形式化的分析结果解读成人们容易理解的信息的过程中都有可能出错，同时在逻辑推理过程中，逻辑推理软件本身也有可能出现错误。更何况由于形式化方法本身的能力以及软件开发过程中提供的信息可能不全面，要在一个系统中完全采用形式化方法几乎是不可能的事情。

4．形式化方法的发展现状

形式化方法的发展之路并非一帆风顺，在它刚被提出的时期，由于符号系统过于艰深，工具也难以上手并且不够完善，人们只在小型实验中采用它。随着形式化方法的不断发展，出现了一些新的方法，如 Z 方法、VDM 方法，既可以严格定义，学习起来也不太困难，并且与之相对应的软件也开发得比较全面，形式化方法才渐渐被真正的软件开发人员所接受。虽然形式化方法目前主要集中在安全性较高的一些领域，但是从实际效果来看，形式化方法确实可以改进软件质量，提高开发效率。正所谓“磨刀不误砍柴工”。虽然采用形

式化方法在开发初期需要更多的时间，但是与在后期所节省的验证维护的人力和时间相比，还是值得的，并且它也确实可以降低软件的错误率，由此受到对安全性能要求高的用户的推崇。

目前已知的采用过形式化方法的软件开发领域包括数据库、医疗、核放射监控、保安系统、通信、交通控制等，均取得了不错的效果。

5．形式化方法小结

形式化方法包含了一组活动和一组模型，它们导致了计算机软件的数学规约。数学规约就是应用一个严格的、数学符号体系来规范、开发和验证软件系统。

形式化方法提供了一种机制，能够克服其他软件工程方法难以克服的二义性、不完整性和不一致性。

形式化方法主要关注软件的功能和数据，而对软件的时序、控制、界面和行为难于表示。

形式化方法不是通过专门的评审与审计来验证软件的正确性，而是通过对软件的数学分析来验证。因此，它能发现和纠正其他方法发现不了的错误。

形式化方法的理论基础是离散数学中的集合运算和逻辑运算，支持形式化方法的基本概念是：

（1）数据不变式。一个在包含一组数据的系统执行过程中总保持为真的条件。

（2）状态。系统访问和修改的存储数据。

（3）操作。系统中发生的动作以及对状态数据的读/写，且一个操作是和两个条件相关联的：前置条件和后置条件。

从国外一些采用形式化方法开发的成功实例可以发现，形式化方法在严密的数学理论的支持下，确实能够解决软件工程中常见的一些问题。随着形式化方法越来越成熟，相关可视化工具也越来越多，将形式化方法引入软件开发的门槛会越来越低。虽然前进的道路很曲折，但是有着独特魅力的形式化方法终会在软件开发中得到越来越广泛的应用。

*1.4.6　面向业务基础平台的方法

业务基础平台（Business Framework）是近几年开始使用的新名词，是 IT 企业开发应用软件的开发环境。近年来，许多软件企业都有自己的业务基础平台。

屏蔽操作系统平台、数据库平台的诸多技术细节，采用面向业务建模来实现软件系统的方法，称为**面向业务基础平台的方法**。

该方法的特点是面向业务领域的与技术无关的开发模式。

面向业务基础平台的开发方法在本质上仍然是面向元数据方法与面向对象方法的综合运用实例，从软件工程方法论的角度来讲，人们并没有将它作为一种单独的基本方法。

该方法的优点：它有效弥合了开发人员和业务人员之间的沟通鸿沟，使开发人员更多地关注业务部分，开发者与用户双方集中精力弄清原始单证与输出报表之间的关系，建立好系统业务模型，而不是关注实现的技术细节，从而提升了业务基础平台中构件的复用性，避免了开发人员开发相同构件的重复劳动，最终达到提高软件开发速度与改进软件产品质量的目的。

该方法的缺点：业务基础平台是面向业务行业领域的，不同行业领域之间的通用业务平台标准尚未建立，也较难建立。因此，不同行业领域的软件开发商，可能有各自不同的业务基础平台。

【例 1-8】　北京某公司在业务基础平台的开发方面，处于国内领先水平。名称为 X3 的业务基础平台，在许多大型企业与软件公司得到了应用，取得了良好业绩。

X3 业务基础平台是从信息化的整体、全局数据库和发展的角度出发，为保障信息化成功而提供的战略