工程应用软件开发技术

唐任仲 编著

化学工业出版社

工程应用软件开发技术

唐任仲　编著

化学工业出版社
·北　京·

（京）新登字 039 号

图书在版编目（CIP）数据

工程应用软件开发技术/唐任仲编著．—北京：化学工业出版社，1999.5
ISBN 7-5025-2548-3

Ⅰ．工… Ⅱ．唐… Ⅲ．软件工程 Ⅳ．TP311.5

中国版本图书馆 CIP 数据核字（1999）第 14152 号

工程应用软件开发技术

唐任仲 编著

责任编辑：周国庆 李玉晖

责任校对：陶燕华

封面设计：郑小红

*

化学工业出版社出版发行

（北京市朝阳区惠新里 3 号 邮政编码 100029）

新华书店北京发行所经销

北京市彩桥印刷厂印刷

北京市彩桥印刷厂装订

*

开本 787×1092 毫米 1/16 印张 12 字数 293 千字

1999 年 5 月第 1 版 1999 年 5 月北京第 1 次印刷

印 数：1—3500

ISBN 7-5025-2548-3/TP・230

定 价：25.00 元

前　言

自1968年在原西德召开的一次有关计算机软件的NATO会议上提出软件工程概念以来，人们已经逐步认识到了软件开发过程不是一个简单的编程过程，而是一个包含了从计划、分析、设计、编码、测试到维护的一系列工程活动的复杂过程。并且，由于软件开发工具功能和性能的日趋完善，以及朝着越来越实用化、可视化和集成化方向的发展，现在开发一个软件系统，甚至有时可以不编写一句源代码都可以实现，因为有现成的软件构件（积木块）供人们使用，只要知道选用哪些积木块、怎样搭配就行。所以今后软件开发的主要任务是分析、设计和维护。目前已经有很多人认识到了这一点，而且也在试图按照软件工程的理论和方法去做。但是，面对着工程中各种各样的实际问题，人们常常感到难于着手进行分析和设计等工作。本书的目的就是企图为读者提供一套实际可操作的软件开发方法和手段。

全书共分9章。首先在第1章中讨论软件及软件开发的基本概念和过程的基础上，分别在第2、3、4章和第5、6、7章中通过结合一个实例详细讨论了如何应用结构化方法和面向对象方法对工程实际问题进行分析描述、软件设计和实现；接着在第8章中讨论了如何应用专家系统方法开发实现能够模拟人类思维方法的软件系统；最后，在第9章中全面例示了应用结构化方法、面向对象方法和专家系统方法解决一个工程实际问题的软件分析、设计和实现过程。

本书是根据作者多年来从事软件开发方面的科研和教学工作的经验体会，在经过多次修改的教学讲义的基础上编写而成的。刘宇、罗忠祥、湛国辉、沙庆丽、鲁青松和潘再其等参加了本书的部分编写和讨论工作，陈寿椿、范新琥、王燮臣、刘国威、吴旭辉和何志民等先生也参加了部分讨论工作，在此表示感谢。

作者衷心感谢陈子辰教授、程耀东教授、潘晓弘副教授、周晓军教授、柯映林教授、顾新建副教授、项占琴副教授、狄瑞坤副教授等对本书的关心和支持。

限于作者的水平，书中一定有不少缺点和错误，恳请读者批评指正。

唐任仲

1999年春节

于杭州求是园

内 容 提 要

本书是根据作者多年来在工程应用软件的研究开发和教学工作中的体会，结合应用实例编写而成的。本书能使读者从工程角度出发对工程应用软件的基本概念、原理、要求、开发方法和手段有一个全面深入的了解，从而能够结合应用单位的实际情况和要求研制开发有关计算机应用软件系统。

全书共分9章。第1章讨论介绍软件及软件开发的基本概念和过程。第2章～第8章分别结合实例详细讨论了软件开发的结构化方法、面向对象方法和专家系统方法的具体分析、设计和实现手段及步骤。第9章为应用实例，比较完整地描述了应用三种软件开发方法完成一个工程实际问题的软件开发过程。本书在内容上并不是简单地将三种软件开发方法罗列出来，而是尽可能地将它们进行有机的融合，在编写上注重实用性和可操作性。

本书可供工程领域计算机应用软件开发人员和应用人员参考，也可作为大专院校相关课程教材或教学参考书。

目　　录

第 1 章　软件开发的基本概念

1.1　软件

根据《GB/T 11457—89 软件工程术语》中的定义，软件是指与计算机系统的操作有关的计算机程序、规程、规则以及任何与之有关的文件。简单地说，软件包括程序和文档两部分。程序是指适合于计算机处理的指令序列以及所处理的数据；文档是与软件开发、维护和使用有关的文字材料。

从不同的角度出发，对软件可以进行不同的分类。例如，按功能划分，可将软件分为系统软件、支撑软件和应用软件；按规模划分，可分为微型、小型、中型、大型及特大型软件等；按工作方式分，可分为实时处理软件、交互式工作软件、分时工作软件等；按服务对象分，可分为仅供一个或少数几个用户使用的项目软件和提供给市场或为成千上万个用户服务的产品软件；还可以按使用频度、失效影响程度等进行划分。上述划分方法主要是从使用者或开发者的角度出发的，如果从计算机本身的处理能力方面出发，则可将软件分为数值计算型软件、逻辑（符号）推理型软件、人机交互型软件和数据密集型应用软件等。

判断一个软件的好坏，是没有什么绝对标准的，但下面给出的一些定性的准则，可以帮助我们理解和判断什么样的软件更好一些。

（1）正确性

正确性是指软件符合规定的需求的程度。正确的软件具备且仅具备软件“规格说明”中所列举的全部功能，能够在预期的环境下完成规定的工作。软件运行的背景条件是否正确，不是正确性考核的范畴。

（2）可靠性

可靠性指的是在规定的条件和时间内软件不引起系统失效的概率。它主要取决于正确性和健壮性两个方面。正确性如前所述；健壮性则是指系统万一遇到意外时能按照某种预定的方式作出适当处理，从而避免出现灾难性的后果。因此，可靠的软件在正常情况下能够正常工作，在意外情况下亦能适当地处理以使软件故障可能导致的损失最小。

（3）简明性

简明性是要求软件简明易读，它和软件设计语言的表达能力以及软件设计风格有关。好的软件设计风格有助于软件达到简明性要求。简明性不等于简单性。问题本来就很复杂时我们不可能使它简单。但软件结构清晰，编排得体，容易看懂还是容易做得到的。最重要的是不要人为地增加复杂性。

（4）有效性

有效性是指软件的时间效率和空间效率要高。随着计算机硬件的快速发展，对于一般软件而言，有效性已不成什么问题，然而对于一些特殊的软件（如实时控制软件）仍是必须认真考虑的。

（5）可维护性

可维护性指的是软件能够修改和升级的容易程度。它目前已经成为越来越重要的软件开

发准则。好的可维护性要求软件有好的可读性、可修改性和可测试性。

(6) 适应性

适应性是指软件使不同的系统约束条件和用户需求得到满足的容易程度。它要求软件尽可能适应各种硬、软件运行环境，以便软件的推广和移植。

一般说来，对于不同的软件，上述准则的优先次序也是各不相同的。对于正规的较大型应用软件，优先次序是正确性、可靠性、可维护性、适应性、简明性、有效性；对于重要的实时控制软件，优先次序是正确性、可靠性、有效性、可维护性、简明性；对于一般应用软件，优先次序为正确性、可靠性、简明性、可维护性、有效性；对于临时软件，则只要求正确性、简明性。

1.2 软件开发

软件开发是一个把用户需要转化为软件需求，把软件需求转化为软件设计，用软件代码来实现软件设计，对软件代码进行测试，并签署确认它可以投入运行使用的过程。在这个过程中的每一阶段，都包含有相应的文档编制工作。

软件是一种产品，具有与其他产品一样的特性。但是，与其他产品相比，软件是一种逻辑的而不是物理的系统成分。在软件开发过程中，它不像加工一个机械零件那样看得见、摸得着。由于不存在物理上的损伤和磨损用坏等问题，所以在软件的开发过程中，人们往往不易或不愿意像开发机器产品、房屋建筑产品那样有计划、有步骤、按规范进行。直至现在，还常常有人喜欢按照自己的一套来“编程序”，拿到一个软件开发课题后，在没有搞好需求分析、结构设计等工作的情况下，就急急忙忙动手编起程序来；由于急于求成，编写程序时也往往忽略好的编码风格，这些都给以后的软件维护工作带来很大的困难。他们习惯于我行我素，不肯学习和采用经过实践证明是行之有效的软件开发方法，有时甚至对这些好的方法采取“抵制”态度。单枪匹马、自以为是、孤芳自赏仍然是我国软件开发工作中存在的严重问题，也是导致我国软件产品水平长期上不去的重要原因之一。

在软件开发过程中，还存在的一个普遍的问题是不重视作为软件的一个重要组成部分的文档编制工作。常常有人认为，软件项目成功的标志是交出能够正确运行的程序，文档是可有可无的。如果一定需要，也只是在程序本身完成之后再补上。这种仅仅为了交差才补写的文档往往和实际开发的程序存在很大差距，难以发挥其应有的作用。符合要求的、规范化的文档在软件开发中的作用就如同零件图纸在产品开发中的作用一样，起着表达思想、传递信息的重要作用，是保证软件开发质量、提高软件可维护性、可靠性和可生产性的重要保障。

1.3 软件开发过程

从工程学角度出发，软件开发过程包括计划、分析、设计、编码、测试和维护等几个阶段，如图 1.1 所示。

(1) 计划

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三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作出估计。制订完成开发任务的实施计划。

(2) 分析

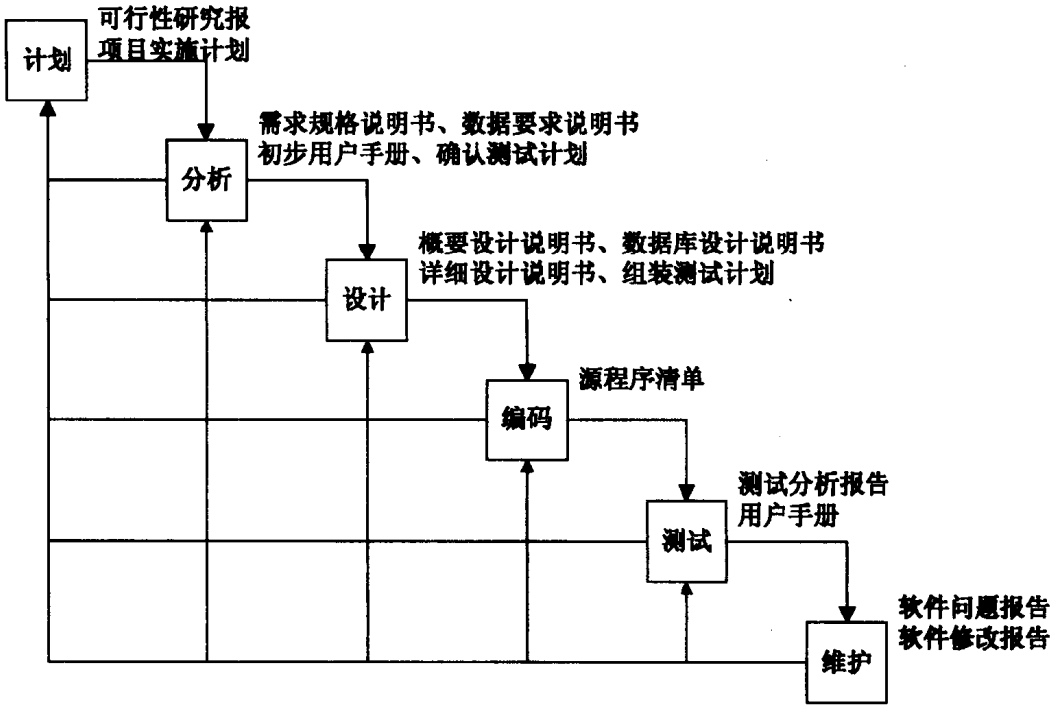

图 1.1　软件开发的迭代模型

对所要解决的问题进行详细定义，这需要软件开发人员和用户共同讨论决定，哪些需求是可以满足的，并加以确切地描述。编写软件需求说明书、初步用户手册、确认测试计划和数据要求说明书等。

（3）设计

根据需求说明书的要求，设计建立相应的软件系统的体系结构，并将整个系统分解成若干个子系统或模块，定义子系统或模块间的接口关系，对各子系统进行具体设计定义。编写软件概要设计和详细设计说明书，数据库或数据结构设计说明书，组装测试计划。

（4）编码

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”。

（5）测试

在设计测试用例的基础上对软件进行测试，以验证系统是否按所要求的性能和功能设想运行。编写测试分析报告。

（6）维护

根据软件运行情况，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。编写软件问题报告、软件修改报告。

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。在软件开发中，重复经常出现。在测试过程中的问题可能要求修改设计，用户可能会提出一些需要修改需求说明书的新要求，等等。

1.4　软件开发原理

在软件开发过程中主要采用以下基本原理。

（1）抽象

计算机只能进行数字、符号和逻辑运算等等，而不能直接处理现实世界中的问题。因此，必须先将要处理的问题按一定的方式和步骤抽象成计算机能处理的形式后再交给计算机去处理。

抽象是具有层次性的。在不同的层次上对问题进行抽象，可以在不同层次上去认识和处理问题。无论是在较高层次上处理问题还是在较低层次上处理问题，都具有同等的重要性。

例如，当利用计算机解决一个较大的问题时，用户会在功能、用户界面等方面提出一些总的要求（高层次上的抽象），同时也会在某些细节上提出一些具体的要求（低层次上的抽象）。在采用结构化方法研制其软件系统时，首先将从满足用户总的要求出发进行系统的总体设计（在较高层次上处理问题），然后进行详细设计以满足用户的具体要求（在较低层次上处理问题）。显然总体设计和详细设计具有同等的重要性。

（2）目标分解

任何一个复杂的问题，都可以通过一些较小的问题表示，这些较小的问题又可以通过更小的问题表示。处理一个复杂的问题，也就可以通过处理那些较小的以及更小的问题来实现。因此，我们可以把处理一个复杂的问题这样一个总目标分解成处理那些较小的以及更小的问题这样一些子目标。

（3）局部化与信息隐藏

局部化简单地说就是尽可能在局部范围内处理好问题。例如，某大学宿舍里有了问题，最好在宿舍内部解决好，而不要弄到班上去解决，而班上有了问题，最好能在班上这个局部范围内解决，而不要弄到系里去。软件开发，也要采用类似的方法。一段程序，它的控制应尽量是局部的，不受其他段的影响，也不影响其他段；所处理的数据也应尽量是局部的。这样，局部定义的数据外部无法访问，达到了信息隐藏的目的。

（4）一致性

一致性是指在整个软件中，所有表示方法应是一致的。例如，同一变量名在整个软件中的意义应该是一致的。遵循一致性原理将大大改善程序的可读性和可维护性，将体现出软件开发者的某种风格。

（5）可验证性

可验证性原理就是所开发的程序模块应尽量具有相对的独立性，能相对独立地测试，相对独立地维护修改，这样能保证整个程序的可验证性。

1.5 软件开发方法

当前主要采用的软件开发方法有结构化方法、面向对象方法和专家系统方法。

1.5.1 结构化方法

结构化方法是一种围绕功能来组织软件系统的方法。在这种方法中，系统的基本构成要素是模块，它是一种实现系统某一功能的程序单元。模块具有输入、输出、内部数据和过程等基本特性。

输入和输出分别是模块需要的和产生的数据，内部数据是仅供模块本身引用的数据，过程则是对模块具体处理细节的描述和表示。输入和输出是模块的外部特性，内部数据和过程是模块的内部特性。

结构化方法是通过按功能将问题分解抽象成模块、建立模块和模块之间的调用关系来进

行软件开发的。参照图 1.1 所示软件开发的一般过程，结构化软件开发方法的具体工作内容和步骤如图 1.2 所示。

分析：问题是什么	设计：怎样解决	实现：解决
分析确定数据流图（DFD）	设计建立结构图（SC）	过程设计
分析确定实体联系图（ERD）	设计建立关系数据模式（RM）	编码调试
		运行维护

图 1.2　结构化软件开发方法工作内容和步骤

1.5.2　面向对象方法

面向对象方法是一种围绕真实世界中的事物来组织软件系统的全新方法。在这种方法中，系统的基本构成要素是对象。从软件开发人员的角度来看，对象是一种将数据和处理这些数据的操作合并在一起的程序单元；从用户的角度来看，对象是一种具有某些属性和行为的事物。对象可以是具体的，如自行车；也可以是概念性的，如车辆通行方案。对象具有标识唯一性、分类性、多态性、继承性和封装性等基本特性。

（1）标识唯一性

面向对象的软件系统是由许许多多的对象构成的，每个对象都有自己的唯一标识，即使两个对象的所有属性和行为都相同，例如两部一模一样的自行车，在面向对象的软件系统中仍然可以区分出是两个不同的对象。通过这种标识，可以找到和确定相应的对象。

（2）分类性

分类性是指将具有相似属性和行为的一组对象抽象成类。在日常生活中，我们说某样东西，例如钟表，一般情况下是指钟表这一类东西，除非特地指明某只具体的钟表。在将对象抽象成类时，往往只是保留与应用有关的重要特性，而忽略那些无关的性质。可以这么说，类是一组在应用领域里具有相同类型属性和行为的对象。

（3）多态性

多态性是指同一操作可以是多个不同的对象的行为。例如操作“move”，可以是自行车对象的行为，也可以是窗口对象的行为。操作是对象的动作或者是对象状态的变化。对象中的操作的具体实现称为方法。由于对象的操作具有多态性，所以它可以有多种实现方法。

（4）继承性

继承性是指对具有层次关系的类的数据和操作进行共享的一种方式。可以先定义一个基本类，然后再在这个基本类的基础上定义多个子类。各个子类继承该基本类（称为父类）的各种性质，并且还具有自己独有的性质。父类的性质在子类中不必重复定义。例如，男式自行车和女式自行车都是自行车的子类，它们都继承了自行车的性质。继承性是面向对象方法的主要优点之一。

（5）封装性

封装性是指将对象的各种独立的外部特性与对象的内部实现细节分离开来，外部特性允许其他对象访问，而内部细节则对其他对象隐藏。具体实现手段是，将表征事物属性的数据以及表征事物行为的操作同放于一对象中，并使对数据的访问只可通过该对象本身的操作来进行，而其他对象都不能直接访问。

面向对象方法是通过将存在于问题空间范围内的事物抽象成对象、建立对象和对象之间的通讯联系来进行软件开发的。参照图 1.1 所示软件开发的一般过程，面向对象软件开发方法的具体工作内容和步骤如图 1.3 所示。

分析：问题是什么	设计：怎样解决	实现：解决
分析确定对象	设计建立问题空间类	操作设计
分析确定结构	设计建立用户界面类	编码调试
分析确定主题	设计建立任务管理类	运行维护
分析确定属性和实例关联	设计建立数据管理类	
分析确定行为和消息关联		

图 1.3　面向对象软件开发方法工作内容和步骤

1.5.3　专家系统方法

专家系统方法是一种围绕知识来组织软件系统的方法。它应用人工智能技术，根据人类专家提供的知识、经验进行推理和判断，模拟人类专家解决那些需要专家解决的复杂问题。在这种方法中，构成系统的基本要素是知识和应用这些知识的推理机制。

专家系统方法是通过对存在于问题空间范围内的知识和经验进行收集、整理和描述，建立知识和知识之间的逻辑推理关系来实现软件开发的。参照图 1.1 所示软件开发的一般过程，专家系统软件开发方法的具体工作内容和步骤如图 1.4 所示。

分析：问题是什么	设计：怎样解决	实现：解决
分析确定知识及其之间关系	设计建立知识库	编码调试
	设计建立推理机	运行维护
	设计建立数据库	
	设计建立解释器	
	设计建立知识获取器	

图 1.4　专家系统软件开发方法工作内容和步骤

1.5.4　各种方法特点比较

从概念方面看，结构化软件是功能的集合，通过模块以及模块和模块之间的分层调用关系实现；面向对象软件是事物的集合，通过对象以及对象和对象之间的通讯联系实现；专家系统软件是知识的集合，通过知识以及知识和知识之间的逻辑推理关系实现。

从构成方面看，结构化软件＝过程＋数据，以过程为中心；面向对象软件＝（数据＋相应操作）的封装，以数据为中心；专家系统软件＝知识＋推理，以知识为中心。

从运行控制方面看，结构化软件采用顺序处理方式，由过程驱动控制；面向对象软件采用交互式、并行处理方式，由消息驱动控制；专家系统软件采用交互式、并行处理方式，由数据驱动控制。

从开发方面看，结构化方法的工作重点是设计；面向对象方法的工作重点是分析；专家系统方法的工作重点是知识的获取与表达。但是，在结构化方法中，分析阶段和设计阶段采用了不相吻合的表达方式，需要把在分析阶段采用的具有网络特征的数据流图转换为设计阶段采用的具有分层特征的结构图，在面向对象方法中则不存在这一问题。

从应用方面看，相对而言，结构化方法更加适合数据类型比较简单的数值计算和数据统计管理软件的开发；面向对象方法更加适合大型复杂的人机交互式软件和数据统计管理软件的开发；专家系统方法更加适合逻辑推理型软件的开发。

从发展方面看，面向对象方法是软件开发方法的发展方向。

第 2 章　结构化分析技术

软件是利用计算机技术解决现实生活中的问题的一种有效方法和手段，如同其他方法和手段一样，首先必须搞清楚要解决的问题是什么，然后才能去解决它。过去总是有人忽略需求分析的重要性，在没有全面、准确和认真地完成需求分析工作之前，就急急忙忙地进行设计甚至实现工作，结果往往是事倍功半，造成不必要的多次反复，甚至给软件留下严重的后遗症。

软件需求分析阶段的工作和任务是在对问题进行调查了解的基础上，用一定的方法和手段对问题进行分析建模。需求分析的结果应该反映的是必须干什么，而不是怎么干。它的主要用途是明确需求、为用户和开发人员提供一起协商讨论的基础、作为设计和实现的依据。

2.1　工作内容和任务

结构化软件开发方法采用结构化分析（Structured Analysis，简称 SA）技术对问题进行分析建模，它将问题表述为

数据流图＋实体联系图

的形式。其中，数据流图描述问题空间中数据变换处理之间的逻辑关系，实体联系图描述问题空间中数据存贮之间的逻辑关系。同时，借用数据词典、结构化语言、判定表、判定树等工具对它们进行详细说明。

因此，结构化分析工作主要包括分析确定数据流图和分析确定实体联系图，具体工作内容和步骤如图 2.1 所示。

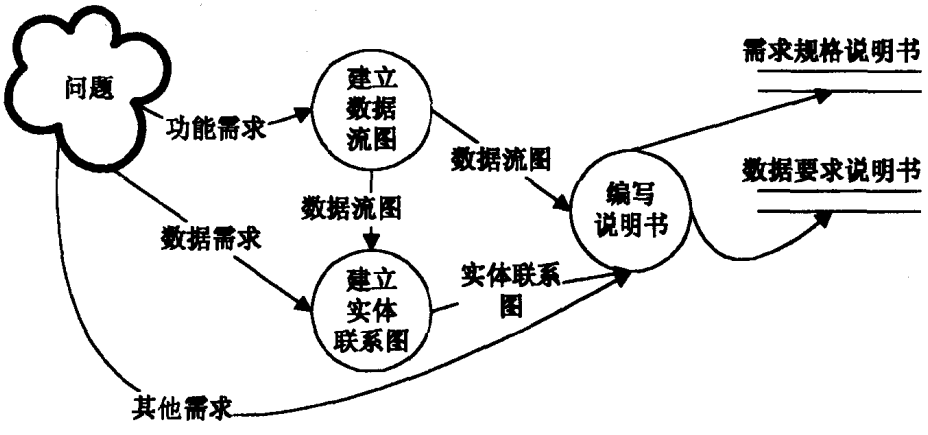

图 2.1　结构化分析工作内容和步骤

首先，根据对问题的功能需求方面的调查了解分析建立数据流图；然后，根据对问题的数据需求方面的了解以及数据流图分析建立实体联系图；最后，根据数据流图、实体联系图以及对问题的性能、资源、可靠性、安全和保密、开发费用、开发进度等其他方面的需求，按照有关规范编写需求规格说明书和数据需求说明书并进行复审，完成对问题的结构化分析建模。

2.2 数据流图

如前所述，进行软件开发要做的第一件事是把所要处理的问题定义清楚，这一工作需要用户和软件开发者（分析员）共同完成。在大多数情况下，用户对计算机是不太熟悉的，因此在进行这一工作时要尽量避免使用计算机行话，尽量避免谈及具体的计算机处理手段和方法，而只需把要处理的问题的内部逻辑关系描述清楚即可。因此，软件需求分析过程也可以称作是软件的逻辑结构设计过程。

例如，对于大学教务管理问题，可用图 2.2 对它进行分析描述：

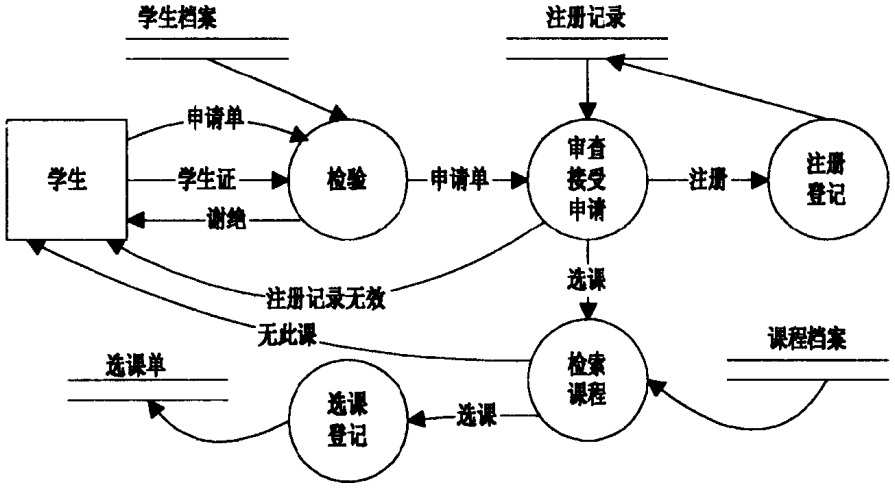

图 2.2 大学教务管理问题数据流图

首先接受学生提交的学生证和申请单，根据学生档案情况检验学生证的有效性和申请单是否填写得正确，如果有误则谢绝，如果无误则根据学生的注册记录情况审查接收学生的申请要求。如果是申请注册，则进行注册登记；如果是申请选课，则先查询课程档案是否有满足申请要求的课程，如果有则进行选课登记，如果没有则通知学生无此课。

上述图形基本上描述清楚了大学教务管理问题中的数据运动情况，这样一个图简明易懂，对于一般的用户来说是很容易接受的。这种图就是所谓的数据流图。

2.2.1 基本数据流图

数据流图（Data Flow Diagram，简称 DFD）是一种最常用的结构化分析工具，它从数据传递和加工角度，以图形的方式刻画系统内的数据运动情况。

数据流图中具有四种基本成分，如图 2.3 所示。

数据流表示数据的流动情况；加工表示对数据的加工处理过程，它的名字应能简明扼要地表明所完成的是什么加工；数据存贮在数据流图中起着保存数据的作用，指向数据存贮的数据流可以理解为写数据，从数据存贮引出的数据流可以理解为读数据，双向数据流可以理解为修改数据；数据源点或终点，表示图中出现数据的始发点或终止点，它在图中的出现仅仅是一种符号，并不需要以软件的形式进行设计和实现。

在数据流图中，如果有两个以上数据流指向一个加工或从一个加工中引出，则这些数据流之间往往存在一定的关系。我们通常用图 2.4 所示符号表示这种关系。

在画数据流图时，如下几个问题值得注意：

（1）是画数据流图而不是画程序框图