第 5 章 结构化分析与设计

结构化分析与设计方法是一种面向数据流的传统软件开发方法，它以数据流为中心构建软件的分析模型和设计模型。结构化分析（structured analysis，SA）、结构化设计（structured design，SD）和结构化程序设计（structured programming，SP）构成了完整的结构化方法。

本章详细介绍结构化分析与设计方法，并通过实例介绍这种方法的实际应用。

5.1 结构化分析方法概述

20 世纪 60 年代末到 20 世纪 70 年代初，Yourdon 等人在“结构化设计”的研究中提出一种表示数据及对数据进行加工变换的图形符号，从而形成了结构化分析方法的雏形。1979 年 De Marco 在他的著作《结构化分析和系统规约》中引入并命名了创建信息流模型的图形符号，提出了使用这些符号的模型。以后，一些学者又陆续提出了结构化方法的一些变种。到 20 世纪 80 年代中期，Ward 和 Mellor 以及 Hatley 和 Pirbhai 对结构化方法进行了扩展，以适应于实时系统的分析[15]。

1. 抽象和分解

“抽象”和“分解”是处理任何复杂问题的两个基本手段。

抽象是指忽略一个问题中与当前目标无关的那些方面，以便更充分地关注与当前目标有关的方面。在求解一个复杂问题时，可以有许多抽象级别。例如，欲用计算机解决一个复杂的应用问题，开发人员首先将该应用问题抽象成一个计算机软件系统。在这个抽象层次上，可以忽略应用问题内部的复杂性，只关注整个软件系统与外界的联系，即软件系统的输入和输出。然后，将这个大而复杂的问题分解成若干个较小的问题（如子系统或功能），每个较小的问题又可分解成若干个更小的问题（如功能或子功能）。如此自顶向下一层一层地分解下去，直至每个最底层的问题都足够简单为止，如图 5.1 所示[16]。这样，一个复杂的问题也就迎刃而解了。

结构化方法就是采用这种自顶向下逐层分解的思想进行分析建模的，自顶向下逐层分解充分体现了分解和抽象的原则。随着分解层次的增加，抽象的级别也越来越低，即越来越接近问题的解。在图 5.1 中，自顶向下的过程是分解的过程，自底向上的过程是抽象的过程。

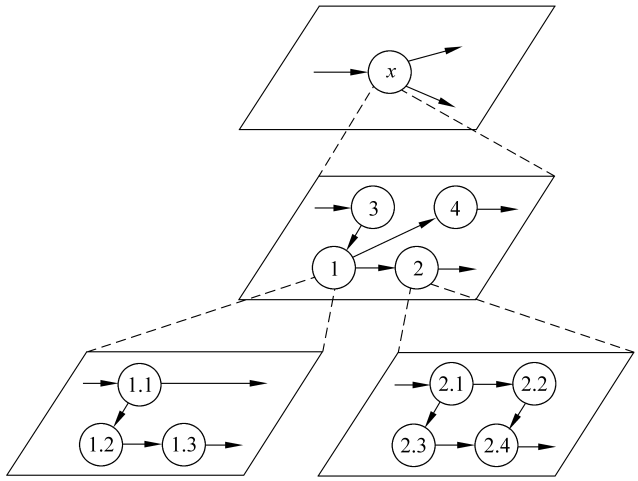

图 5.1　抽象与分解

2. 结构化分析的过程

结构化分析的过程可以分为如下 4 个步骤[16]：

① 理解当前的现实环境，获得当前系统的具体模型(物理模型)。

② 从当前系统的具体模型抽象出当前系统的逻辑模型。

③ 分析目标系统与当前系统逻辑上的差别，建立目标系统的逻辑模型。

④ 为目标系统的逻辑模型作补充。

3. 结构化分析模型的描述形式

结构化分析方法导出的分析模型采用图 5.2 所示的描述形式[15]。

图 5.2　结构化分析模型的结构

图 5.2 中，数据字典是模型的核心，包括对软件使用和产生的所有数据的描述。围绕数据字典有 3 种图以及相应的规约(specification)或描述。

数据流图用于软件系统的功能建模，描述系统的输入数据流如何经过一系列的加工，逐步变换成系统的输出数据流，这些对数据流的加工实际上反映了系统的某种功能或子功能。数据流图中的数据流、文件、数据项、加工都应在数据字典中描述。加工规约是对数据流图中的加工的说明，在结构化方法中用加工的“小说明”作为加工规约。

实体-关系图(E-R 图)用于数据建模，描述数据字典中数据之间的关系。数据对象的属性用“数据对象描述”来描述，通常存放在数据字典中。

状态转换图用于行为建模，描述系统接收哪些外部事件，以及在外部事件的作用下系统的状态迁移(即从一个状态迁移到另一个状态)。控制规约用来描述软件控制方面的附加信息。

结构化分析方法的分析结果包括：一套分层的数据流图、一本数据字典(包括 E-R 图)、一组加工规约以及其他补充材料(如非功能性需求等)。

5.2 数据流图

数据流图(data flow diagram,DFD)描述输入数据流到输出数据流的变换(即加工),用于对系统的功能建模。

5.2.1 数据流图的图形表示

本节介绍数据流图的基本图形元素及其扩充符号。

1. 数据流图的基本图形元素

数据流图中的基本图形元素包括：数据流、加工、文件、源或宿。其中,数据流、加工、文件用于构建软件系统内部的数据处理模型;源或宿表示存在于系统之外的对象,以帮助我们理解系统数据的来源和去向。DFD 的基本图形元素如图 5.3 所示。

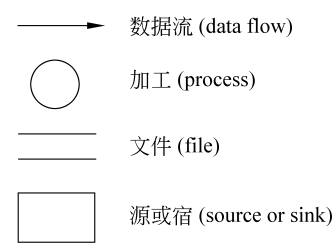

图 5.3 DFD 的基本图形元素

需要说明的是,DFD 图形元素还可以用其他描述符号来表示,如用圆角矩形表示加工,用开放箭头表示数据流。

(1) 源或宿

源或宿通常是指存在于软件系统之外的人员或组织,表示软件系统输入数据的来源和输出数据的去向,因此也称为源点和终点。例如,对一个考务处理系统而言,考生向系统提供报名单(输入数据流),所以,考生是考务处理系统(软件)的一个源;而考务处理系统要将考试成绩的统计分析表(输出数据流)传递给考试中心,所以,考试中心是该系统的一个宿。

在许多系统中,某个源和某个宿可以是同一个人员或组织,此时,在 DFD 中可以用同一个符号表示。例如,考生向系统提供报名单(输入数据流),而系统向考生送出准考证(输出数据流),所以,在考务处理系统中,考生既是源又是宿。

源和宿用相同的图形符号表示。当数据流从该符号流出时,表示它是源;当数据流流向该符号时,表示它是宿;当两者皆有时,表示它既是源又是宿。

(2) 加工

加工描述了输入数据流到输出数据流的变换,即将输入数据流加工成输出数据流。每个加工用一个定义明确的名字标识。一个加工可以有多个输入数据流和多个输出数据流,但至少有一个输入数据流和一个输出数据流。例如,考务处理系统中可以有统计成绩、编准考证号、审定合格者等加工。

(3) 数据流

数据流由一组固定成分的数据组成。例如,运动会管理系统中,报名单(数据流)由队名、姓名、性别、参赛项目等数据组成。

在 DFD 中,数据流的流向可以有以下几种：从一个加工流向另一个加工,从加工流向文件(写文件),从文件流向加工(读文件),从源流向加工,从加工流向宿。

DFD中的每个数据流用一个定义明确的名字标识。然而,对于流向文件或从文件流出的数据流,由于它们代表了文件的一个记录,所以不必为它们命名。

值得注意的是,在DFD中描述的是数据流,而不是控制流。区分数据流和控制流的方法是看流中包含的信息是数据还是控制信号或控制条件。

(4) 文件

文件用于存放数据。通常一个流入加工的数据流经过加工处理后就消失了,而它的某些数据(或全部数据)可能被加工成输出数据流,流向其他加工或宿。除此之外,在软件系统中还常常要把某些信息保存下来供以后使用,此时可使用文件。例如,考务处理系统中,报名时产生的考生名册要随着报名的过程不断补充,在统计成绩和制作考生通知书时还要使用考生名册的相关信息。因此,考生名册可作为文件存在,以保存相关的考生信息。

每个文件用一个定义明确的名字标识。可以有数据流流入文件,表示写文件;也可以有数据流从文件流出,表示读文件;也可以用双向箭头的数据流指向文件,表示对文件的修改。

这里要说明的是,DFD中的文件在具体实现时可以用文件系统实现,也可以用数据库系统来实现。文件的存储介质可以是磁盘、磁带或其他存储介质。

图5.4给出一个简化的图书订购系统的数据流图。

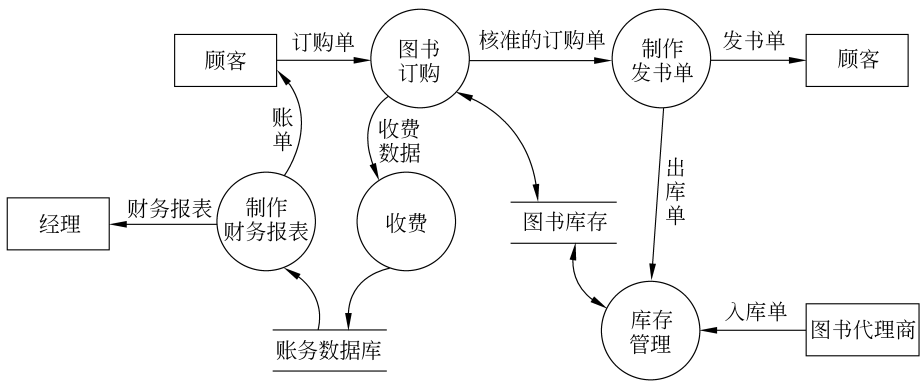

图5.4 简化的图书订购系统的DFD

2. 数据流图的扩充符号

在DFD中,一个加工可以有多个输入数据流和多个输出数据流,此时可以加上一些扩充符号来描述多个数据流之间的关系。

(1) 星号(*)

星号表示数据流之间存在“与”关系。如果是输入流则表示所有输入数据流全部到达后才能进行加工处理;如果是输出流则表示加工结束后将同时产生所有的输出数据流。

(2) 加号(+)

加号表示数据流之间存在“或”关系。如果是输入流则表示其中任何一个输入数据流到达后就能进行加工处理;如果是输出流则表示加工处理的结果至少产生其中一个输出数据流。

(3) 异或(⊕)

异或号表示数据流之间存在“异或”(互斥)关系。如果是输入流则表示当且仅当其中一个输入流到达后才能进行加工处理;如果是输出流则表示加工处理的结果仅产生这些输出数据流中的一个。

3. 数据流图的层次结构

从原理上讲,只要有足够大的纸,一个软件系统的分析模型可以画在一张图上。然而,一个复杂的软件系统可能涉及到几百个加工和几百个数据流,甚至更多。如果将它们画在一张图上,则会十分复杂,不易阅读,也不易理解。

George Miller 在著名的论文《神奇的数字 7 加减 2:我们处理信息的能力的某种限制》中指出:“人们在一段时间内的短期记忆似乎限制在 5～9 件事情之内(除非此人学会使用联想记忆法的技巧)。”

根据自顶向下逐层分解的思想,可以将数据流图画成如图 5.1 所示的层次结构,每张图中的加工个数可大致控制在“7 加减 2”的范围中,从而构成一套分层数据流图。

(1) 层次结构

分层数据流图的顶层只有一张图,其中只有一个加工,代表整个软件系统,该加工描述了软件系统与外界(源或宿)之间的数据流,称为顶层图。顶层图中的加工(即系统)经分解后的图称为 0 层图,也只有一张。处于分层数据流图最底层的图称为底层图,在底层图中,所有的加工不再进行分解。分层数据流图中的其他图称为中间层图,其中至少有一个加工(也可以是所有加工)被分解成一张子图。在整套分层数据流图中,凡是不再分解成子图的加工称为基本加工。

(2) 图和加工的编号

在介绍图和加工的编号之前,先介绍父图和子图的概念。

如果某图(记为 A)中的某一个加工分解成一张子图(记为 B),则称 A 是 B 的父图,B 是 A 的子图。若父图中有 n 个加工,则它可以有 0～n 张子图,但每张子图只对应一张父图。

为了方便对图的管理和查找,可以采用下列方式对 DFD 中的图和加工编号:

- 顶层图只有一个加工(代表整个软件系统),该加工不必编号。
- 0 层图中的加工编号分别为 1、2、3…。
- 对于子图号,若父图中的加工号 x 分解成某一子图,则该子图号记为“图 x”。
- 对于子图中加工的编号,若父图中的加工号为 x 的加工分解成某一子图,则该子图中的加工编号分别为 $x.1$、$x.2$、$x.3$…。

例如,加工 2.3.4 的子图号为“图 2.3.4”,图 2.3.4 中的加工号分别为 2.3.4.1、2.3.4.2、2.3.4.3…。对于层次较多的 DFD,其较低层的加工号会很长,因此,在图中,可先标上图号,其中的加工号可简写为.1、.2、.3…。

这种编号方式可方便地根据图号将分层数据流图整理成册,并可方便地根据加工号查到相关的子图,也可方便地从子图找到其对应的父图。

5.2.2 分层数据流图的画法

本节以一道软件专业技术资格和水平考试(简称为资格和水平考试)高级程序员级(下午考试)试题为例,介绍分层数据流图的画法。该试题的背景是资格和水平考试的考务处理系统(已作简化),该考试的目的是认定考生具备的软件技术资格和水平,分成多个级别,如初级程序员、程序员、高级程序员、系统分析员等,凡满足一定条件的考生都可参加某一级别的考试。由于每年的试题难度很难保持完全的一致,因此考试的合格标准将根据每年的考试成绩由考试中心确定。考试的阅卷由阅卷站进行,因此,阅卷工作不包含在该软件系统中。

该考务处理系统的功能需求(工作过程)说明如下:

- 对考生提交的报名单进行检查。
- 对合格的报名单编好准考证号后将准考证送给考生,并将汇总后的考生名单送给阅卷站。
- 对阅卷站送来的成绩清单进行检查,并根据考试中心制定的合格标准审定合格者。
- 制作考生通知单并发放给考生。
- 进行成绩分类统计(按地区、文化程度、职业、考试级别等分类)和试题难度分析,产生统计分析表。

部分数据流和文件的组成如下所示:

报名单=地区+序号+姓名+文化程度+职业+考试级别+通信地址

正式报名单=准考证号+报名单

准考证=地区+序号+姓名+准考证号+考试级别+考场

考生名单={准考证号+考试级别} (其中{w}表示 w 重复多次)

考生名册=正式报名单

统计分析表=分类统计表+难度分析表

考生通知单=准考证号+姓名+通信地址+考试级别+考试成绩+合格标志

下面介绍画分层数据流图的步骤。

1. 画出系统的输入和输出

系统的输入和输出用顶层图来描述,即描述系统从外部的哪些源接收哪些输入数据流,以及系统的哪些输出数据流送往外部的哪些宿。

(1) 确定源或宿

分析系统的功能说明,系统外部的人或组织有考生、阅卷站和考试中心,他们都向系统提供信息(数据流)并接收系统输出的信息(数据流),因此,他们都既是源又是宿。

(2) 确定加工

顶层图只有一个加工,即软件系统,根据本例的实际含义可取名为考务处理系统。

(3) 确定数据流

顶层图中的数据流就是系统的输入输出信息。分析系统的功能说明,可以确定以下输入数据流和输出数据流。

输入数据流:报名单(来自考生)、成绩清单(来自阅卷站)、合格标准(来自考试中心)。

输出数据流：准考证(送往考生)、考生名单(送往阅卷站)、考生通知书(送往考生)、统计分析表(送往考试中心)。

有经验的软件工程师还会想到，软件系统应该对输入数据进行合法性检查，以提高系统的健壮性，避免错误数据对系统造成不良的后果。例如，报名单填写得不完整，成绩清单中各小题得分超出该题得分范围等。因此系统还应增加两个输出数据流：不合格报名单(返回给考生)，错误成绩清单(返回给阅卷站)。由于合格标准比较简单，这里就不返回错误的合格标准了(必要时可要求重新输入)。

这两个新增加的输出数据流都是考虑提高系统健壮性而加入的，在画顶层图时可以先不考虑，等到画0层图时可以发现需要添加这些属于系统的数据流，此时，再回过头来将其添加到顶层图中。

(4) 顶层图通常没有文件

根据以上分析，可画出该系统的顶层图，如图5.5所示。

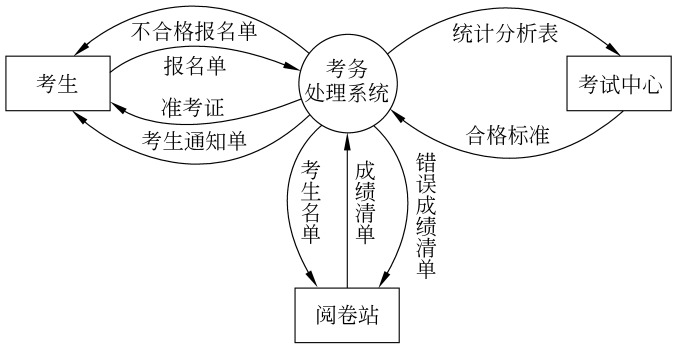

图5.5　考务处理系统顶层图

2. 画出系统内部

将顶层图中的加工(即系统)分解成若干个子加工，并用一些新定义的数据流进行连接，使得系统的输入数据流(即顶层图的输入数据流)经过一连串的加工处理后，变换成系统的输出数据流(即顶层图的输出数据流)。这个图即为0层图。

下面介绍确定加工、数据流、文件、源或宿的一般方法，该方法可适用于绘制0层图和所有其他子图。

(1) 确定加工

这里讲的加工是指父图中某加工分解而成的子加工。通常，可以用下列两种方法来确定加工。

① 根据功能分解来确定加工

一个加工实际上反映了系统的一种功能，根据功能分解的原理，可以将一个复杂的功能分解成若干个较小的功能，每个较小的功能就是分解后的子加工。这种方法较多应用于高层DFD中加工的分解。

② 根据业务处理流程确定加工

分析父图中即将分解的加工的业务处理流程，业务流程中的每一步都可能是一个子加

工。特别要注意在业务流程中数据流发生变化或数据流的值发生变化的地方,应该存在一个加工,该加工将原数据流(作为该加工的输入数据流)加工处理成变化后的数据流(作为该加工的输出数据流)。例如,考务处理系统中,考生提交的报名单在核准后,系统应给考生一个准考证号,并将其添加到报名单中(称为正式报名单),以使准考证号与相应的考生关联。此时在核准的报名单(合格报名单)与正式报名单之间就存在一个加工,该加工的功能是编准考证号,如图5.6所示。这种方法较多应用于低层DFD中加工的分解,它能描述父加工中输入数据流到输出数据流之间的加工细节。

(2) 确定数据流

当用户把若干数据作为一个单位来处理(即一起到达,一起加工)时,则把这些数据看作一个数据流[16]。通常,实际工作环境中的表单就是一种数据流,如报名单、日报表等。

在父图中某加工分解而成的子图中,父图中相应加工的输入输出数据流就是子图边界上的输入输出数据流。另外,在分解后的子加工之间应增添一些新的数据流,这些数据流是加工过程中的中间数据(对某子加工输入数据流的改变),它们与所有子加工一起完成了父图中相应加工的输入数据流到输出数据流的变换。如果某些中间数据需要保存,以备以后使用,那么可以表示为流向文件的数据流。

同一个源或加工可以有多个数据流流向另一个加工,如果它们不是一起到达和一起加工的,那么可以将它们分成多个数据流。例如,在银行自动取款机(ATM)上取钱,客户(源)向"读取银行卡信息"(加工)提供的信息有:与银行卡相关的数据(如卡号等,通过划卡获取)和密码(通过人工录入)。由于它们不是同时到达和同时加工的,所以应看作两个数据流,如图5.7所示。

图5.6 根据数据流的变化确定加工

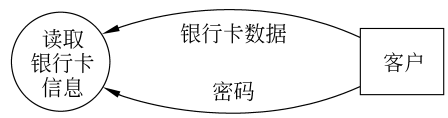

图5.7 多数据流

同样,同一个加工也可以有多个数据流流向另一个加工或宿。

(3) 确定文件

在父图中某加工分解而成的子图中,如果父图中该加工存在流向文件的数据流(写文件),或者存在从文件流向该加工的数据流(读文件),则这种文件和相关的数据流都应画在子图中。

在分解的子图中,如果需要保存某些中间数据,以备以后使用,那么可以将这些数据组成一个新的文件。在自顶向下画分层数据流图时,新文件(首次出现的文件)至少应有一个加工为其写入记录(即从该加工流向文件的数据流),同时至少存在另一个加工读取该文件的记录(即从文件流向加工的数据流)。

注意,对于从父图中继承下来的文件,在子图中可能只对其读记录,或只对其写记录。

(4) 确定源和宿

通常在0层图和其他子图中不必画出源和宿。有时为了提高可读性,可以将顶层图中的源和宿画在0层图中。

当同一个外部实体(人或组织)既是系统的源,又是系统的宿时,可以用同一个图形符号

来表示。为了画图的方便，避免图中线的交叉，同一个源或宿也可以重复画在 DFD 的不同位置，以增加可读性，但他们仍代表同一个实体，如图 5.4 中的“顾客”。

在考务处理系统的 0 层图中，采用功能分解方法来确定加工。分析系统的需求说明，可知系统的功能主要分为考试报名及统计成绩两大部分。其中报名工作在考试前进行，统计成绩工作在考试后进行。

为此，定义两个加工：“考试报名”和“统计成绩”。0 层图中的数据流，除了继承了顶层图中的输入数据流和输出数据流外，还应定义这两个加工之间的数据流。由于这两个加工分别在考试前后进行，并不存在直接关系，因此“考试报名”所产生的结果“考生名册”应作为文件保存，以便考试后由“统计成绩”读取。于是，该考务系统的 0 层图如图 5.8 所示。

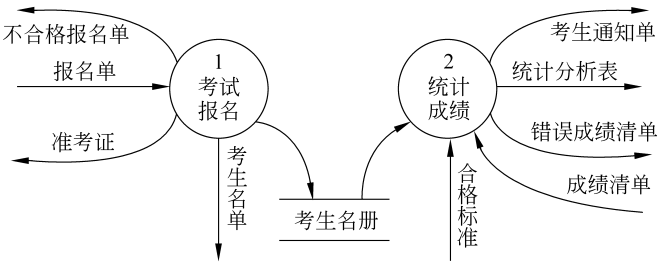

图 5.8　考务处理系统 0 层图

3. 画出加工内部

当 DFD 中存在某个比较复杂的加工时，可以将它分解成一张 DFD 子图。分解的方法是：将该加工看作一个小系统，该加工的输入输出数据流就是这个假设的小系统的输入输出数据流，然后采用画 0 层图的方法，画出该加工的子图。

下面介绍考务处理系统 0 层图中加工 1 的分解。这里根据业务处理流程来确定由加工 1 分解而成的子加工。分析考务处理系统功能需求说明和 0 层图，其中与加工 1(考试报名)相关的业务流程是：首先检查考生送来的报名单，然后编准考证号，并产生准考证，最后产生考生名单和考生名册(文件)。因此，可以将加工 1 分解成 3 个子加工：检查报名单，编准考证号，登记考生。

“检查报名单”加工的功能是，接收考生提交的“报名单”，当检查未通过时产生“不合格报名单”；如通过检查，则产生“合格报名单”。“编准考证号”加工的功能是，为具有合格报名单的考生编准考证号，将“准考证”发送给考生，并将准考证号添加到合格报名单中，形成“正式报名单”。“登记考生”加工的功能是，根据正式报名单制作“考生名册”文件和“考生名单”。其中“合格报名单”和“正式报名单”是新增加的内部数据流，其他数据流都是加工 1 原来就有的。

在加工 1 的分解中没有新的文件产生。

根据上述分析，可以画出加工 1 分解而成的子图，如图 5.9 所示。

图 5.9 中，虽然“报名单”、“合格报名单”和“不合格报名单”的数据组成是相同的，但它们的性质不同，所以用 3 个不同的名字。“正式报名单”是对“合格报名单”的修改(添加了“准考证号”)，所以这两个数据流之间必定有一个加工(即“编准考证号”)。

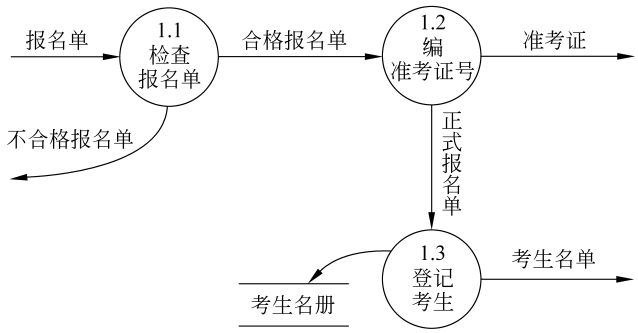

图 5.9　考务处理系统加工 1 子图

可以用同样的方法画出加工 2 分解的 DFD 子图,如图 5.10 所示。由于加工“分类统计成绩”和“分析试题难度”是在整个考试工作后期做的,所以,要将“正确成绩清单”保存为“试题得分清单”文件。

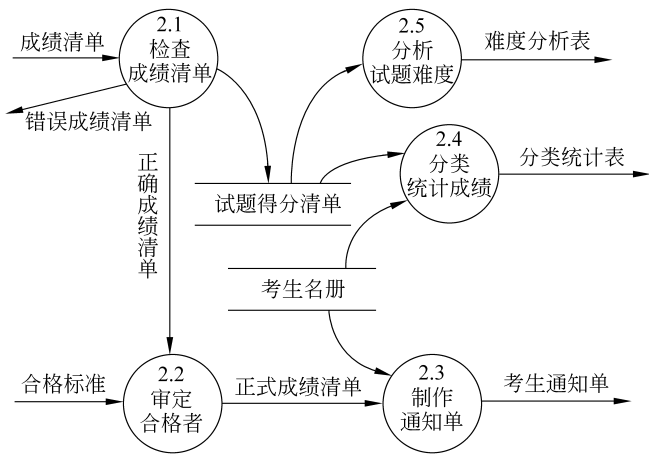

图 5.10　考务处理系统加工 2 子图

4. 重复第 3 步,直至每个尚未分解的加工都足够简单

这里假定图 5.9 和图 5.10 中的每个加工都已足够简单(即不必再分解),该考务处理系统的分层 DFD 的绘制工作结束。

5.3　分层数据流图的审查

在结构化分析过程中,构造分层 DFD 与建立数据字典的工作常常是交替进行的。在后面的介绍中可以看到组成数据流数据的不同会导致 DFD 的不同。因此,在画 DFD 时,应该同时给出每个数据流和文件的组成。有关数据流和文件组成的描述见 5.4 节。

分层数据流图画好后,应该认真检查图中是否存在错误,或不合理(不理想)的部分。本节从分层 DFD 的一致性和完整性、构造分层 DFD 时需注意的问题以及分解程度等几个方面,来说明如何审查分层 DFD 的合理性。