第5章　结构化系统设计

学习目标和指南

学习目标：

1. 从总体了解系统设计的主要任务和设计步骤，学会编制系统设计文档。
2. 了解系统物理配置方案设计的内容以及软硬件选择的原则。
3. 理解模块间的关系，掌握模块化设计方法。
4. 理解代码设计的作用，了解代码的种类，掌握代码的设计方法和校验方法。
5. 明确文件设计的步骤和方法，掌握数据库设计方法。
6. 掌握输入/输出的设计方法，包括设计的内容格式、设备的选择。
7. 掌握流程设计的步骤及流程设计方法。

学习指南：

1. 从总体上了解系统设计的主要任务和系统总体设计的内容。牢固掌握结构化设计和模块化设计的思想和模块调用方式。

2. 模块结构图是描述系统模块结构的图形工具，理解模块、模块间的耦合、模块内的聚合等概念，掌握模块化设计方法。

3. 掌握代码的作用，了解代码的种类、设计方法和设计原则，了解代码校验位的作用和其确定、校验步骤。

4. 输入/输出设计一定程度上是一种艺术性工作，因此学习过程中，不仅对设计内容要求把握，同时应善于根据不同的信息形态选择合理的设备，学会鉴赏界面。

5. 数据库设计首先要对用户需求进行调查分析，学会将客观世界中的数据转换为信息。

6. 处理流程设计是关系到系统全局的，应从系统分析的逻辑模型出发，结合数据库设计、代码设计和输入/输出设计，得出合理的、流畅的系统全局处理流程图。

课前思考

如果说系统分析阶段所提出的系统逻辑模型解决了“做什么”的问题，那么系统设计阶段解决什么问题呢？

在系统分析阶段，我们明确了新系统的功能结构及信息结构，也就是系统的逻辑模型，对新系统回答了“做什么”的问题。在系统设计阶段我们需要回答的中心问题是“怎么做”，即通过给出新系统物理模型的方式描述如何实现在系统分析中规定的系统功能。

系统设计是信息系统开发过程中的重要阶段。在这一阶段中我们将要根据前一阶段系统分析的结果，在已经获准的系统分析报告的基础上，进行新系统设计。系统设计阶段的主要任务是：在科学、合理的设计和总体模型的基础上，尽可能提高系统的运行效

率、可变性、可控性和工作质量。充分利用并合理投入各类人、财、物资源，使之获得较高的综合效益。

5.1 结构化系统设计概述

5.1.1 系统设计目标和原则

1. 系统设计目标

系统设计的目的是在保证实现逻辑模型功能的基础上，尽可能提高目标系统的简单性、可变性、一致性、完整性、可靠性、经济性、系统的运行效率和安全性，将分析阶段所获得的系统逻辑模型转换成一个具体的计算机实现方案的物理模型，包括计算机物理系统配置方案报告和一份系统设计说明书。

2. 系统设计原则

(1) 简单性。在达到预定的目标、具备所需要的功能前提下，系统应尽量简单，这样可减少处理费用，提高系统效益，便于实现和管理。

(2) 灵活性和适应性。可变性是现代化企业的特点之一，是指其对外界环境的变化的适应能力。作为企业的管理信息系统必须具有相当的灵活性，以便适应外界环境的不断变化，而且系统本身也需不断修改和改善。因此，在这里系统的可变性是指允许系统被修改和维护的难易程度。一个可变性好的系统，各个部分独立性强，容易进行变动，从而可提高系统的性能，不断满足对系统目标的变化要求。此外，如果一个信息系统的可变性强，可以适应其他类似企业组织的需要，这将比重新开发一个新系统成本要低得多。

(3) 一致性和完整性。一致性是指系统中信息编码、采集、信息通信要具备一致性，设计规范应标准；完整性是指系统作为一个统一的整体而存在，系统功能应尽量完整。

(4) 可靠性。系统的可靠性指系统硬件和软件在运行过程中抵抗异常情况的干扰及保证系统正常工作的能力。衡量系统可靠性的指标是平均故障间隔时间和平均维护时间。前者指平均的前后两次发生故障的时间，反映了系统安全运行时间；后者指故障后平均每次所用的修复时间，反映了系统可维护性的好坏。只有可靠的系统才能保证系统的质量并得到用户的信任，否则就没有使用价值。

提高系统可靠性的途径主要有：

① 选取可靠性较高的主机和外部设备。

② 硬件结构的冗余设计，即在高可靠性的应用场合，应采取双机或双工的结构方案。

③ 对故障的检测处理和系统安全方面的措施，如对输入数据进行校检，建立运行记录和监督跟踪，规定用户的文件使用级别，对重要文件的复制等。

(5) 经济性。系统的经济性是指系统的收益应大于系统支出的总费用。系统支出费用包括系统开发所需投资的费用与系统运行维护费用之和；系统收益除了货币指标外，还有非货币指标。

系统应该给用户带来相应的经济效益。系统的投资和经营费用应当得到补偿。需要指出的是，这种补偿有时是间接的或不能定量计算的。特别是对于管理信息系统，它的效益当中，有很大一部分效益不能以货币来衡量。

5.1.2 系统设计内容

1. 系统总体结构设计

系统总体结构设计包括两方面的内容：数据处理设计和系统物理配置方案设计。

2. 模块结构设计

应用变化分析方法和事务分析方法构建系统的模块结构。

3. 代码设计

代码设计就是通过设计合适的代码形式，使其作为数据的一个组成部分，用以代表客观存在的实体、实物和属性，以保证它的唯一性并便于计算机处理。

4. 数据库设计

根据系统分析得到数据字典，再结合系统处理流程图，进行数据库设计。

5. 输入/输出设计

输入/输出设计主要是对以纪录为单位的各种输入输出报表格式的描述，另外，对人机对话格式的设计和输入/输出装置的考虑也在这一步完成。

6. 处理流程设计

处理流程设计是通过系统处理流程图的形式，将系统对数据处理过程和数据在系统存储介质间的转换情况详细地描述出来。

7. 系统设计文档

系统设计文档指系统设计说明书、程序设计说明书、系统测试说明书以及各种图表等，要将它们汇集成册，交有关人员和部门审核批准。

5.1.3 系统设计的步骤

根据系统设计的内容，可以把系统设计分为两个阶段：总体设计阶段和详细设计阶段。总体设计阶段决定系统的模块结构，而详细设计阶段是具体考虑每一模块内部采用什么算法。具体来说，在总体设计中，根据系统分析的成果进行系统总体结构设计，包括网络结构设计、硬件结构设计、软件结构设计、数据库存储和处理方式设计等。详细设计阶段包括具体的代码设计、输入/输出设计、信息分类和数据库设计、功能模块设计。详细设计是对上述总体设计的结果进行进一步细化，直至符合小组编程的要求。

5.2 系统物理配置方案设计

5.2.1 设计依据

(1) 系统吞吐量,即每秒钟执行的作业数。系统吞吐量越大,则系统的处理能力就越强。系统吞吐量与系统硬、软件的选择有着直接的关系,如果要求系统具有较大的吞吐量,就应当选择具有较高性能的计算机和网络系统。

(2) 系统响应时间,是从用户向系统发出一个作业请求开始,经系统处理后给出应答结果的时间。如果要求系统具有较短的响应时间,就应当选择运算速度较快的 CPU 及具有较高传递速率的通信线路,如实时应用系统。

(3) 系统可靠性,是系统可以连续工作的时间。例如,对于每天需要 24 小时连续工作的系统,可以采用双机双工结构方式。

(4) 集中式 (Centralized Processing) 或分布式 (Distributed Processing)。如果一个系统采用集中式的处理方式,则信息系统既可以是主机系统,也可以是网络系统;如果系统处理方式是分布式的,则应采用微机网络。

(5) 地域范围。对于分布式系统,要根据系统覆盖的范围决定采用广域网还是局域网。

5.2.2 计算机硬件及网络选择

计算机硬件的选择主要取决于数据处理方式和运行的软件系统。管理者对计算机的基本要求是速度快、容量大、通道能力强、操作灵活方便,但计算机的性能越高,价格就越昂贵。一般来说,如果系统的数据处理是集中式的,系统应用的主要目的是利用计算机的强大计算能力,则可以采用主机-终端系统,以大型机或中小型机作为主机。对于企业管理分布式的应用,采用微机网络更为灵活、经济。

应用软件对计算机处理能力的需求主要包括:①计算机内存;②CPU;③输入/输出和通信通道数目;④显示方式;⑤外接转储设备及其类型。

对于计算机网络的选择方面,可以采用网络操作系统,例如 NetWare、Windows NT、UNIX 等。UNIX 历史最早,是唯一能够适用于所有应用平台的网络操作系统。NetWare 适用于文件服务器/工作站模式,具有较高的市场占有率。Windows NT 随着 Windows 操作系统的发展和客户机-服务器模式向浏览器-服务器模式延伸,成为很有发展前景的网络操作系统。

5.2.3 数据库管理系统的选择

管理信息系统是以数据库系统为基础,一个好的数据库管理系统对管理信息系统的应用有着举足轻重的重要影响。在数据库管理系统的选择上,主要考虑:①数据库的性能;②数据库管理系统的系统平台;③数据库管理系统的安全保密性能;④数据的类型。

目前,软件市场上有许多数据库管理系统,例如 Oracle、Sybase、SQL Server、Informix FoxPro 等。Oracle、Sybase 是大型数据库管理系统,运行于客户机-服务器模式,是开发大

型 MIS 的首选；FoxPro 在小型 MIS 中最为流行。

5.2.4 应用软件的选择

根据应用需求来开发管理信息系统最容易满足用户的特殊管理要求，但是成本较高。随着技术成熟、设计规范、管理思想先进的商品化应用软件的推广，系统设计人员就面临着对应用软件的选择问题。如果直接应用商品化软件，既可以节省投资，又能够规范管理过程，加快系统应用的进度，就是说不一定要自行开发，而是可以选用这些成熟的商品化软件。

选择应用软件应考虑以下因素：

(1) 是否能够满足用户的需求。

根据系统分析的结果，在软件功能上应注意以下问题：

① 系统必须处理哪些事件和数据，软件能否满足数据表示的需要。

② 系统能够产生哪些报告、报表、文档或其他输出。

③ 系统要存储的数据量及必须满足哪些查询需求。

(2) 软件的灵活性。

由于存在管理需求上的不确定性，系统应用环境会经常发生变化。因此，应用软件要有足够的灵活性，以适应对软件的输入、输出和系统平台升级要求。

(3) 软件的技术支持。

对于商品化软件，稳定的技术支持是必需的。一方面是为了保证软件能够满足需求的变化，另一方面是便于今后升级。

(4) 相关企业对应用软件的选择情况。

5.3 模块结构设计

5.3.1 模块结构图

模块结构图(或称结构图，Structure Chart)是 1974 年由 W. Steven 等人从结构化设计的角度提出的一种工具。它的基本做法是将系统划分为若干子系统，子系统下再划分为若干的模块，大模块内再分小模块。而模块是指具备输入输出、逻辑功能、运行程序和内部数据四种属性的一组程序。

模块结构图是用于描述系统模块结构的图形工具，它不仅描述了系统的子系统结构与分层的模块结构，还清楚地表示了每个模块的功能，而且直观地反映了块内联系和块间联系等特性。

1. 模块的概念

模块是组成目标系统逻辑模型和物理模型的基本单位，它的特点是可以组合、分解和更换。系统中任何一个处理功能都可以看成是一个模块。根据模块功能具体化程度的不同，可以分为逻辑模块和物理模块。在系统逻辑模型中定义的处理功能可视为逻辑模块。物理

模块是逻辑模块的具体化，可以是一个计算机程序、子程序或若干条程序语句，也可以是人工过程的某项具体工作。

一个模块应具备 4 个要素：

(1) 输入和输出。模块的输入来源和输出去向都是同一个调用者，即一个模块从调用者那里取得输入，进行加工后再把输出返回调用者。

(2) 处理功能。指模块把输入转换成输出所做的工作。

(3) 内部数据。指仅供该模块本身引用的数据。

(4) 程序代码。指用来实现模块功能的程序。

前两个要素是模块的外部特性，即反映了模块的外貌。后两个要素是模块的内部特性。在结构化设计中，主要考虑的是模块的外部特性，其内部特性只做必要了解，具体的实现将在系统实施阶段完成。

2. 模块结构图的基本符号

模块结构图是结构化设计中描述系统模块结构的图形工具。作为一种文档，它必须严格地定义模块的名字、功能和接口，同时还应当在模块结构图上反映出结构化设计的思想。模块结构图由模块、调用、数据、控制信息和转接符号五种基本符号组成，如图 5.1 所示。

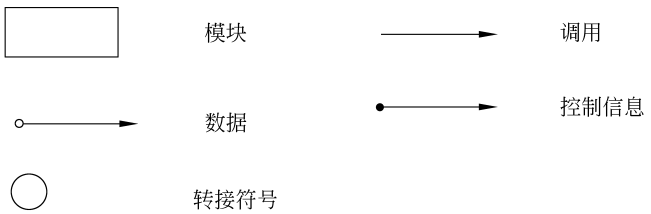

图 5.1　模块结构图的五种基本符号

1) 模块

这里所说的模块通常是指用一个名字就可以调用的一段程序语句。在模块结构图中，用长方形框表示一个模块，长方形中间标上能反映模块处理功能的模块名字。模块名通常由一个动词和一个作为宾语的名词组成。

2) 调用

在模块结构图中，用连接两个模块的箭头表示调用，箭头总是由调用模块指向被调用模块，但是应该理解成被调用模块执行后又返回到调用模块。

如果一个模块是否调用一个从属模块决定于调用模块内部的判断条件，则该调用称为模块间的判断调用，采用菱形符号表示。如果一个模块通过其内部的循环功能来循环调用一个或多个从属模块，则该调用称为循环调用，用弧形箭头表示。判断调用和循环调用的表示方法如图 5.2 所示。

3) 数据

当一个模块调用另一个模块时，调用模块可以把数据传送到被调用模块处供处理，而被调用模块又可以将处理的结果数据送回到调用模块。在模块之间传送的数据，使用与调用

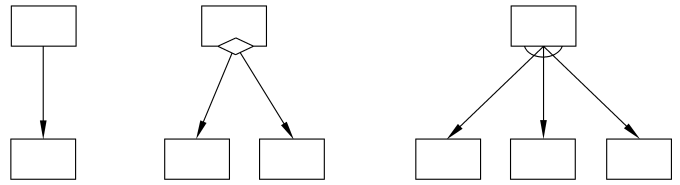

图 5.2　判定调用和循环调用

箭头平行的带空心圆的箭头表示，并在旁边标上数据名。例如，图 5.3(a)表示模块 A 调用模块 B 时，A 将数据 x、y 传送给 B，B 将处理结果数据 z 返回给 A。

4) 控制信息

为了指导程序下一步的执行，模块间有时还必须传送某些控制信息，例如，数据输入完成后给出的结束标志，文件读到末尾所产生的文件结束标志等。控制信息与数据的主要区别是前者只反映数据的某种状态，不必进行处理。在模块结构图中，用带实心圆点的箭头表示控制信息。例如，图 5.3(b)中“无此学生”就是用来表示送来的学生号有误的控制信息。

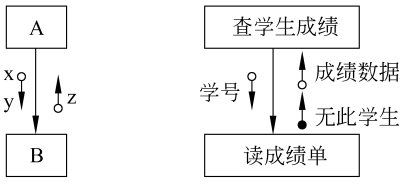

图 5.3　模块调用

5) 转接符号

当模块结构图在一张图面上画不下，需要转接到另外一张纸上，或为了避免图上线条交叉时，都可使用转接符号，圆圈内加上标号。

5.3.2　模块间的关系

1. 模块间耦合

所谓耦合，就是指两个实体相互依赖于对方的一个量度。模块之间联系越紧密，其耦合性就越强，模块的独立性就越差。模块间耦合的高低取决于模块间接口的复杂性、调用的方式以及传递的信息。在计算机系统中，两个不同模块之间往往存在不同程度的耦合。由于需要两个或者多个模块之间相互协同工作，因此计算机系统的耦合部分往往是容易产生 BUG 的部分。因此，在模块耦合度较高的系统之中，往往存在较高的缺陷率。

一般模块之间的耦合关系为：

1) 非直接耦合

如果两个模块之间没有直接关系，它们之间的联系完全是通过主模块的控制和调用来实现的，这就是非直接耦合(Nondirective Coupling)。这种耦合的模块独立性最强。

2) 数据耦合

如果一个模块访问另一个模块时，彼此之间是通过数据参数(不是控制参数、公共数据结构或外部变量)来交换输入、输出信息的，则称这种耦合为数据耦合(Data Coupling)。由于限制了只通过参数表传递数据，数据耦合开发的程序界面简单、安全可靠，因此，数据耦合

是松散的耦合，模块之间的独立性比较强。在软件程序结构中必须有这类耦合。

3）标记耦合

如果一组模块通过参数表传递记录信息，就是标记耦合(Stamp Coupling)。实际上，这组模块共享了这个记录，它是某一数据结构的子结构，而不是简单变量。这些模块都必须清楚该记录的结构，并按结构要求对此记录进行操作。在设计中应尽量避免这种耦合，它使在数据结构上的操作复杂化了。如果采取“信息隐蔽”的方法，把在数据结构上的操作全部集中在一个模块中，就可以消除这种耦合。

4）控制耦合

如果一个模块通过传送开关、标志、名字等控制信息，明显地控制选择另一模块的功能，就是控制耦合(Control Coupling)。这种耦合的实质是在单一接口上选择多功能模块中的某项功能。因此，对所控制模块的任何修改，都会影响控制模块。另外，控制耦合也意味着控制模块必须知道所控制模块内部的一些逻辑关系，这些都会降低模块的独立性。

5）外部耦合

一组模块都访问同一全局简单变量而不是同一全局数据结构，而且不是通过参数表传递该全局变量的信息，则称之为外部耦合(External Coupling)。例如，C 语言程序中各个模块都访问被说明为 extern 类型的外部变量。外部耦合引起的问题类似于公共耦合，区别在于在外部耦合中不存在依赖于一个数据结构内部各项的物理安排。

6）公共耦合

一组模块都访问同一个公共数据环境，则它们之间的耦合就称为公共耦合(Common Coupling)。公共的数据环境可以是全局数据结构、共享的通信区、内存的公共覆盖区等。

2. 模块内聚合

内聚是指一个模块内各个元素彼此结合的紧密程度。内聚按强度从低到高有以下 7 种类型：

(1) 偶然内聚。如果一个模块的各成分之间毫无关系，则称为偶然内聚。

(2) 逻辑内聚。几个逻辑上相关的功能被放在同一模块中，则称为逻辑内聚。如一个模块读取各种不同类型外设的输入。尽管逻辑内聚比偶然内聚合理一些，但逻辑内聚的模块各成分在功能上并无关系，所以局部功能的修改有时也会影响全局，因此这类模块的修改也比较困难。

(3) 时间内聚。如果一个模块完成的功能必须在同一时间内执行(如系统初始化)，但这些功能只是因为时间因素关联在一起，则称为时间内聚。

(4) 过程内聚。如果一个模块内部的处理成分是相关的，而且这些处理必须以特定的次序执行，则称为过程内聚。

(5) 通信内聚。如果一个模块的所有成分都操作同一数据集或生成同一数据集，则称为通信内聚。

(6) 顺序内聚。如果一个模块的各个成分和同一个功能密切相关，而且一个成分的输出作为另一个成分的输入，则称为顺序内聚。

(7) 功能内聚。模块的所有成分对于完成单一的功能都是必需的，则称为功能内聚。

5.3.3 模块化设计方法

在系统分析阶段，我们采用结构化分析方法得到了由数据流图、数据字典和加工说明等组成的系统的逻辑模型。现在，可根据一些规则从数据流图导出系统初始的模块结构图。

管理信息系统的数据流图通常也可分为两种典型的结构，即变换型结构和事务型结构。变换型结构的数据流图呈一种线性状态，如图 5.4 所示，它所描述的工作可表示为输入、主处理及输出。事务型结构的数据流图则呈束状，如图 5.5 所示，即一束数据流平行流入或流出，可能同时有几个事务要求处理。

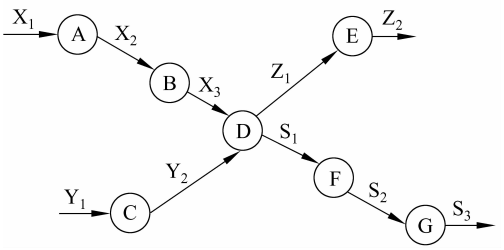

图 5.4 变换型结构的数据流图

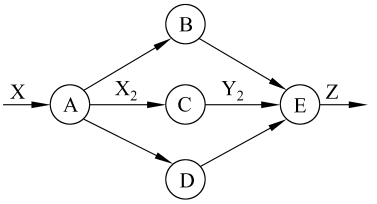

图 5.5 事务型结构的数据流图

这两种典型的结构分别可通过“变换分析”和“事务分析”技术，导出“变换型”和“事务型”初始的模块结构图。这两种方法的思想是首先设计顶层模块，然后自顶向下，逐步细化，最后得到一个满足数据流图所表示的用户要求的系统的模块结构图，即系统的物理模型。

下面分别讨论通过“变换分析”和“事务分析”技术，导出“变换型”和“事务型”初始结构图的技术。

1. 变换分析方法

因为变换型结构由输入、主处理和输出三部分组成，所以从变换型结构的数据流图导出变换型模块结构图，可分三步进行。

1）找出系统的主加工

为了处理方便，先不考虑数据流图中的一些支流，如出错处理等。

通常在数据流图中多股数据流的汇合处往往是系统的主加工。若没有明显的汇合处，则可先确定哪些数据流是逻辑输入和逻辑输出，从而获得主加工。

从物理输入端一步步向系统中间移动，直至到达这样一个数据流，它再不能被作为系统的输入，则其前一个数据流就是系统的逻辑输入，即离物理输入端最远的，但仍可被视为系统输入的那个数据流就是逻辑输入。

用类似方法，从物理输出端一步步向系统中间移动，则离物理输出端最远的，但仍可被视为系统输出的那个数据流就是逻辑输出。

逻辑输入和逻辑输出之间的加工就是我们要找的主加工，如图 5.6 所示。

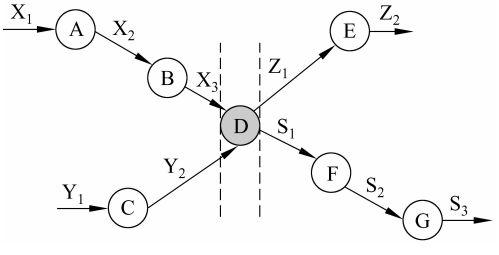

图 5.6 在数据流图中找系统的主加工

2）设计顶层模块和第一层模块

首先在与主加工对应的位置上画出主模块(如图 5.7 所示)，主模块的功能就是整个系统要做的工作，主模块又称为主控制模块。主模块是模块结构图的“顶”，现在我们就可按“自顶向下，逐步细化”的思想来画模块结构图顶下的各层了。每一层均需按输入、变换、输出等分支来处理。模块结构图第一层的画法如下：

(1) 为每一个逻辑输入画一个输入模块，其功能是向主模块提供数据。

(2) 为每一个逻辑输出画一个输出模块，其功能是把主模块提供的数据输出。

(3) 为主处理画一个变换模块，其功能是把逻辑输入变换成逻辑输出。

至此，结构图第一层就完成了。

在作图时应注意主模块与第一层模块之间传送的数据，要与数据流图相对应。

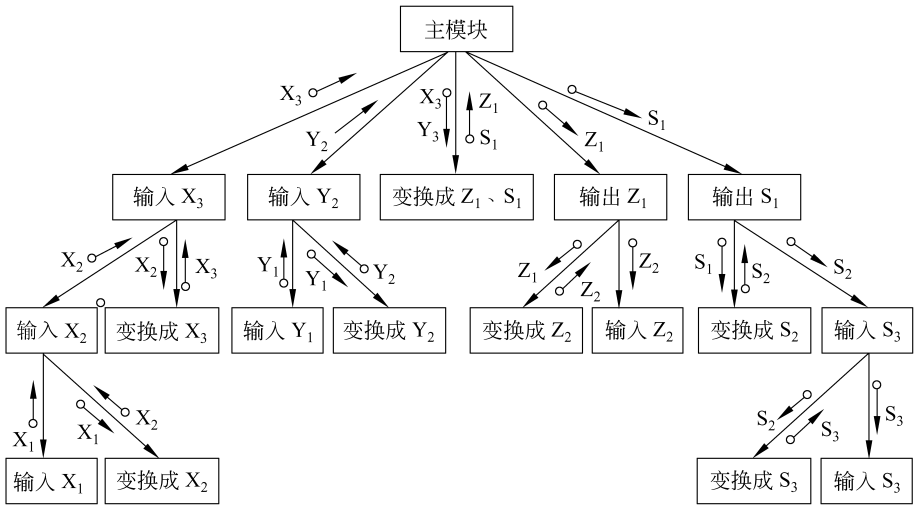

图 5.7　由变换型数据流图导出的初始模块结构图

3）设计中、下层模块

因为输入模块的功能是向调用它的模块提供数据，所以它自己也需要一个数据来源。此外，输入模块必须向调用模块提供所需的数据，因此它应具有变换功能，能够将输入数据按模块的要求进行变换后，再提交该调用模块。从而，我们为每个输入模块设计两个下层模块，其中一个是输入模块，另一个是变换模块。

同理，也为每个输出模块设计两个下层模块，一个是变换模块，将调用模块所提供的数据变换成输出的形式；另一个是输出模块，将变换后的数据输出。

该过程自顶向下递归进行，直到系统的物理输入端或物理输出端为止(如图 5.7 所示)。每设计出一个新模块，应同时给它起一个能反映模块功能的名字。

运用上述方法，就可获得与数据流图相对应的初始结构图。

2. 事务分析方法

当数据流图呈现“束状”结构时，应采用事务分析的设计方法。就步骤而言，该方法与变