# 第 5 章　面向数据流的分析方法

面向数据流的分析方法（dataflow-oriented analysis method）与面向对象、面向数据的分析方法，都是需求建模方法。它们均有一组规范的语言表达机制，需求分析人员用来表达用户需求、构造软件系统模型。此外，它们还含有一些规则和经验知识，指导分析人员提取需求信息，促进用户需求精确化、完全化和一致化。

面向数据流的分析方法是结构化分析方法系列中的一支，具有明显的结构化特征。结构化分析方法的雏形出现于 20 世纪 60 年代后期。但是，直到 1979 年才由 DeMarco 将其作为一种需求分析方法正式提出。由此，结构化分析方法得到了迅速发展和广泛应用。

本章主要介绍广为使用的面向数据流的分析方法及其需求分析 CASE 工具。

## 5.1　数据流图与数据字典

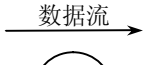
一个基于计算机的信息处理系统就是对数据流进行一系列加工的处理过程，而这些加工将输入数据流变换为输出数据流。数据流图就是用来刻画数据流和加工的信息系统建模技术。数据字典是与数据流图配套使用的，用来定义系统中数据元素的有机集合体。

### 5.1.1　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转换（即加工），用于对系统的功能建模。

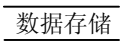
1. 数据流图的基本图形元素

数据流图中的基本图形元素包括：数据流、转换、数据存储以及外部实体，如图 5-1 所示。数据流、转换、数据存储用于构建软件系统内部的数据处理模型；外部实体表示存在于系统边界之外的对象，用来帮助我们理解软件系统数据的来源和去向。



在转换之间定向流动的数据项或数据项集合

变换数据流的处理过程，又称泡（bubble）



为一个或多个转换提供数据源或数据存储服务的缓冲区、文件或数据库

位于软件系统边界之外的信息生产者或消费者

图 5-1　数据流图的基本图形元素

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

（1）外部实体。外部实体通常是指存在于软件系统之外的人员或组织，表示软件系统数据的来源和去向。例如，在考务处理系统中，考生向系统提供报名单，所以考生是考务处理系统的一个外部实体；而系统要将考试成绩的统计分析传递给考试中心，考试中心也是一个外部实体。

（2）转换。转换描述了输入数据流到输出数据流的变换，也就是将输入数据流加工成输出数据流。每个转换用一个定义明确的名字标识。一个转换可以有多个输入或输出数据流，但至少要有一个输入数据流和一个输出数据流。例如。考务处理系统中有审定合格者、编制准考证和统计成绩等转换。

一个转换可以代表一系列程序、单个程序或者程序的一个模块；它甚至可以代表目视检查数据正确性等人工处理过程。

（3）数据流。数据流由一组固定成分的数据组成。例如，考务处理系统向考生送出的准考证由姓名、考场、考号、考试时间和考试科目等数据组成。

DFD 中的每个数据流用一个定义明确的名字标识。对于流入或流出数据存储的数据流，由于代表了数据存储的一个有效内容，所以不必为其命名。

注意，数据流与程序流程图中用箭头表示的控制流有着本质不同，千万不要混淆。熟悉程序流程图的初学者在画数据流图时，往往试图在数据流图中表现分支条件或循环，殊不知这样做将造成混乱，导致画不出正确的数据流图。在数据流图中应该描绘所有可能的数据流向，而不是描绘出现某个数据流的条件。

（4）数据存储。在软件系统中常常把某些信息保存起来供以后使用，这在 DFD 中用数据存储来表示。例如，在考务处理系统中，考生名册要随着报名的过程不断补充。因此，考生名册可以作为一个数据存储。

DFD 中的每个数据存储用一个定义明确的名字标识。可以有流入数据流，代表写操作；也可以有流出数据流，代表读操作。

一个数据存储可以表示一个文件、文件的一部分、数据库的元素或记录的一部分等；数据可以存储在磁盘、主存及其他任何介质上(包括人脑)。

数据存储和数据流都是数据，仅仅所处的状态不同。数据存储是处于静止状态的数据，数据流则是处于运动中的数据。

2. 绘制数据流图

（1）顶级 DFD。顶级数据流图只有一个转换，代表整个软件系统，主要描述了软件系统与外界（外部实体）之间的数据流。例如，图 5-2 是“家庭保安系统”的顶级数据流图。

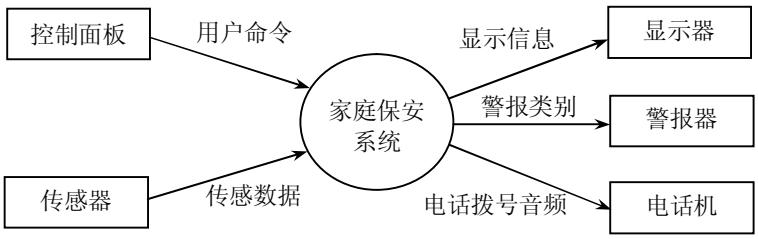



图 5-2 “家庭保安系统”顶级数据流图

（2）逐层分解。数据流图提供了层次结构，让分析人员能够方便地表示任意抽象级别上的信息系统或其子系统，并支持问题分解、逐步求精的分析方法，充分体现了分解和抽象的原则。

初始时，整个信息处理系统可以用图 5-2 所示的顶级（第 0 级）数据流图表示。随着需求分析活动的逐渐深入，较高抽象级别上的复杂加工可以精化为一系列相互关联的数据流和子加工，如图 5-3 所示。这种自上向下、逐层分解的过程一直进行到所有的加工都可以清晰、明确的定义为止，从而构成了整套分层数据流图。

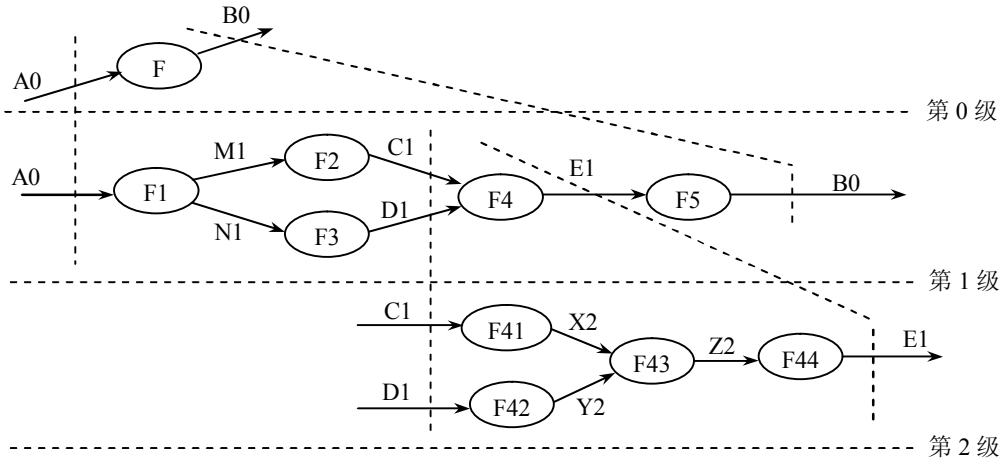



图 5-3　数据流图的精化与层间平衡

在数据流方法中，对数据（数据流）的精化是伴随着对加工的逐步精化而同步进行的。

（3）原则。建立数据流模型要遵循以下原则。

1）每个加工至少应有一个输入数据流（反映被处理数据的来源）和一个输出数据流（反映加工的结果）。

2）数据流图中各构成元素的名称必须具有明确的含义且能够代表对应元素的内容或功能。

3）对数据流图中某个加工进行细化生成的下层数据流图，称为其上层图的子图。应保证分层数据流图中任意对应的父图和子图的输入/输出数据流的一致性，如图 5-3 所示。

4）在数据流图中，应按照层次给每个加工编号，用于表明该加工所处的层次及上下层的父图与子图的关系。编号的规则为：顶层加工不用编号；第一层加工的编号为 1，2，…，n。第二层加工的编号为 11，12，…，21，22，…，n1，n2，…，等，依此类推。

5）在父图中不要出现子图中涉及的局部数据存储文件。通常除底层数据流图中需表明所有数据存储外，为保持画面整洁，各中间层数据流图只需显示处于加工之间的接口文件即可。

6）数据流图只能由 4 种基本符号组成，是实际业务流程的客观映像，用于说明系统应该“做什么”，而不需要指明系统“如何做”。

7）数据流图的分解速度应保持适中。通常一个加工每次可分解为 2～4 个子加工，最多不要超过 7 个，否则会增加用户理解的难度。同时要注意，逐层精化必须适可而止。

8）为了便于数据流图在计算机上的输入和输出，应避免使用斜线、弧线、圆等符号。

3. 标识数据流图的元素

数据流图中每个成分的命名是否恰当，直接影响数据流图的可理解性。因此，给这些成分起名字时应该仔细推敲。下面讲述在命名时应注意的问题。

（1）为数据流（或数据存储）命名。

1）名字应代表整个数据流（或数据存储）的内容，而不是仅仅反映它的某些成分。

2）不要使用空洞的、缺乏具体含义的名字（如“数据”、“信息”、“输入”之类）。

3）如果在为某个数据流（或数据存储）起名字时遇到了困难，则很可能是因为对数据流图分解不恰当造成的，应该试试重新分解，看是否能克服这个困难。

（2）为加工命名。

1）通常先为数据流命名，然后再为与之相关联的加工命名。这样命名比较容易，而且体现了人类习惯的“由表及里”的思考过程。

2）名字应该反映整个加工的功能，而不是它的一部分功能。

3）名字最好由一个具体的及物动词，加上一个具体的宾语组成。应尽量避免使用“加工”、“处理”等空洞笼统的动词作名字。

4）通常名字中仅包括一个动词。如果必须用两个动词才能描述整个加工的功能，则把这个加工再分解成两个加工可能更恰当些。

5）如果在为某个加工命名时遇到困难，则很可能是发现了分解不当的迹象，应考虑重新分解。

外部实体并不需要在开发目标系统的过程中设计和实现，它并不属于数据流图的核心内容，只不过是目标系统的外围环境部分（可能是人员、外部设备或传感器等）。通常，为外部实体命名时采用它们在问题域中习惯使用的名字（如“报警器”、“控制面板”等）。

### 5.1.2 数据字典

前述的数据流图机制并不足以完整地描述软件需求，因为它没有描述数据流和数据源的内容。因此，需要一种系统化的方式来描述每个数据对象的特性，数据字典正是用来完成这项任务的。数据流图必须与描述并组织数据条目的数据字典配套使用。

数据字典是在结构化分析过程中定义对象的内容时，使用的一种半形式化的工具。下面是对这个重要的建模工具的定义。

数据字典是所有与系统相关的数据元素的有组织的列表，并且包含了对这些数据元素的精确、严格的定义。其作用是使得用户和系统分析员双方对输入、输出、存储的成分甚至中间加工结果有共同的理解。简而言之，数据字典是对系统中的所有数据元素的定义集合。

目前，数据字典作为 CASE“结构化分析与设计工具”的一部分实现。尽管不同工具中数据字典的形式不同，但是数据字典一般应包含下列信息。

（1）名字——数据、控制项、数据存储或外部实体的主要名称。

（2）别名——第一项中所列诸对象的其他名字。

（3）使用地点与方式——使用数据或控制项的加工的列表，以及使用这些对象的方式（例如，作为加工的输入，从加工输出，作为数据存储，作为外部实体）。

（4）内容描述——描述数据或控制项内容的符号。

（5）补充信息——关于数据类型、预置值、限制等的其他信息。

一旦把数据对象或控制项的名字和别名输进数据字典，就可以保持命名的一致性。也就是说，支持数据字典的 CASE 工具能够发现重名现象并发出警告信息，这提高了分析模型的一致性，有助于减少错误。

“使用地点与方式”信息是从数据流图中自动提取的。表面看起来，数据字典工具的这项功能好像并不重要，实际上这正是数据字典的主要优点之一。在分析过程中几乎始终在进行修改。但对于大型项目来说，确定修改的影响往往很困难。许多软件工程师都遇到过下述问题：这个数据对象在什么地方使用？如果修改了它，相应地还应该再修改哪些对象？这个改动在整体上有什么影响？利用数据字典中的“使用地点与方式”信息，完全可以回答上述问题。

下面介绍用于书写“内容描述”信息的符号，也就是定义数据的方法。

定义绝大多数复杂事物，都是用被定义事物成分的某种组合来表示，这些组成成分又由更低层的成分的组合来定义。从这个意义上说，定义就是自顶向下的分解。所以数据字典中的定义，就是对数据自顶向下的分解。那么，应该把数据分解到什么程度呢？一般说来，当分解到不需要进一步定义，每个和工程有关的人也都清楚其含义的元素时，这种分解过程就完成了。

由数据元素组成数据的方式只有下述三种基本类型。

（1）顺序——即以确定次序连接两个或多个分量。

（2）选择——即从两个或多个可能的元素中选取一个。

（3）重复——即把指定的分量重复零次或多次。

因此，可以使用上述三种关系算符定义数据字典中的任何条目。为了说明重复次数，重复算符通常和重复次数的上下限同时使用（当上下限相同时表示重复次数固定）。当重复的上下限分别为 1 和 0 时，可以用重复算符表示某个分量是可选的（可有可无的）。但是，“可选”是由数据元素组成数据时一种常见的方式，把它单独列为一种算符可以使数据字典更清晰一些。因此，增加了下述的第四种关系算符：

（4）可选——即一个分量是可有可无的（重复零次或一次）。

虽然可以使用自然语言描述由数据元素组成数据的关系，但是为了更加清晰简洁起见，建议在数据字典中采用如表 5-1 所示的基本符号。

**表 5-1　数据字典中的基本符号及其含义**

| 符号 | 含义 | 说明 |
|---|---|---|
| = | 表示定义为 | 用于对=左边的条目进行确切的定义 |
| + | 表示与关系 | X=a+b 表示 X 由 a 和 b 共同构成 |
| [ \| ] 或[ , ] | 表示或关系 | X=[a\|b]与 X=[a,b]等价，表示 X 由 a 或 b 组成 |
| ( ) | 表示可选项 | X=(a)表示 a 可以在 X 中出现，也可以不出现 |
| { } | 表示重复 | 大括号中的内容重复 0 到多次 |
| m{ }n | 表示规定次数的重复 | 重复的次数最少 m 次，最多 n 次 |
| “ ” | 表示基本数据元素 | “ ”中的内容是基本数据元素，不可再分 |
| .. | 连接符 | Month=1..12 表示 month 可取 1～12 中的任意值 |
| * * | 表示注释 | 两个星号之间的内容为注释信息 |

对于大型软件项目来说，数据字典的规模十分庞大，人工管理将非常困难。目前大多数支持数据流分析的 CASE 工具都具备数据字典管理功能，这些功能包括：

（1）一般性检查。当分析人员要求创建新的数据条目并键入名称或别名时，CASE 工具自动进行重名检查，这就避免了数据流图中不一致的数据定义。

（2）CASE 工具可根据已有的数据流图生成相关加工的列表。并且，随着数据流图的进化，CASE 工具可自动修改该列表，以便数据字典和数据流图在任何时刻都保持一致。

（3）CASE 工具将自动完成有关数据条目的各种查询，例如：该数据条目在何处使用？修改某一部分数据流图将会影响哪些数据条目？修改某数据条目又会造成哪些影响？显然，对这些问题的正确回答将有助于分析人员在需求模型的进化过程中维持模型的一致性。

数据条目的定义必须遵循以下原则：精确、简洁，并且能为用户方和软件开发方共同理解。可以使用形式语言中的语法定义机制描述数据条目的内容。原子语法成分则用简单明了的自然语言予以描述。例如，“家庭保安系统”中的“电话号码”数据条目可以定义如下：

<电话号码>=<分机号>|<外线号码>

<分机号>=1816|1817|…|1858

<外线号码>=9+（<市话号码>|<长话号码>）

<长话号码>=0+（<区号>+<市话号码>）

<区号>=*任何长度为 3 的数字串*

<市话号码>=<局号>+<分局号>

<局号>=395|396|397|303|304|305

<分局号>=*任何长度为 4 的数字串*

综上所述，利用数据字典可以对数据流图中的数据流、数据源以及外部实体进行描述、组织和管理。同时，对于加工，也需要一种比图形记号更为详尽的表示机制，这就是结构化的文字描述（参考 7.2.6 节）。分析人员可以在数据流图的任一加工上附加一段文字，用以说明加工的功能、性能要求及设计约束等，这种说明应尽可能简洁、清晰、易于理解。

## 5.2 实体—关系图

在数据密集型应用问题中，对复杂数据及数据之间复杂关系的分析和建模将成为需求分析的重要任务。很显然，这项任务是简单的数据字典机制无法胜任的。所以，有必要在数据流分析方法中引进适合于复杂数据建模的工具：实体—关系图。

### 5.2.1 数据对象、属性与关系

1. 数据对象

数据对象是现实世界中实体的数据表现。或者说，数据对象是现实世界中省略了功能和行为的实体。在数据流分析方法中，数据对象包括 5.1 节提及的数据存储、数据流以及外部实体的数据部分。例如，在汽车销售管理问题中，“制造商”、“汽车”以及“经销商”都是数据对象。

数据对象仅仅封装了数据而没有对作用于数据上的操作的引用，这是数据对象与面向对象范型中的“类”或“对象”的显著区别。

2. 属性

数据对象的性质由其属性刻画。通常属性包括：

（1）命名性属性：对数据对象的实例命名，其中必含有一个或一组关键属性，以便唯一地标识数据对象的实例。同时，把一个或多个属性定义为“标识符”，也就是当我们希望找到数据对象的一个实例时，标识符属性就成为“关键字”。

（2）描述性属性：对数据对象实例的性质进行刻画。

（3）引用性属性：将自身与其他数据对象的实例关联起来。

一般而言，现实世界中任何给定实体都具有许多属性，分析人员应当并且只能考虑与应用问题有关的属性。例如，在汽车销售管理问题中，汽车的属性可能有：制造商、型号、标识码、车体类型、颜色和买主等。

3. 关系

应用问题中的任何数据对象都不是孤立的，它们与其他数据对象一定存在各种形式的关联。例如，在汽车销售管理问题中，“制造商”与“汽车”之间存在“生产”关系，“购车者”与“汽车”之间存在“购买”关系。当然，关系的命名及内涵因具体问题而异。分析人员必须善于剔除与应用问题无关的关系。

关系也称为联系。联系可分为以下三类。

（1）一对一联系（1:1）。

例如，一个部门有一个经理，而每个经理只在一个部门任职，则部门与经理的联系是一对一的。

（2）一对多联系（1:N）。

例如，某校教师与课程之间存在一对多的联系“教”，即每位教师可以教多门课程，但是每门课程只能由一位教师来教。

（3）多对多联系（M:N）。

例如，表示学生与课程间的联系（“学”）是多对多的，即一个学生可以学多门课程，而每门课程可以有多个学生来学。

联系也可能有属性。例如，学生“学”某门课程所取得的成绩，既不是学生的属性也不是课程的属性。由于“成绩”既依赖于某名特定的学生又依赖于某门特定的课程，所以这是学生与课程之间的联系“学”的属性。

基于数据对象、属性与关系，分析人员可以为应用问题建立数据模型。为确保模型的一致性并消除数据冗余，分析人员要掌握以下规范化规则：

（1）数据对象的任何实例对每个属性必须有且仅有一个属性值。

（2）属性是原子数据项，不能包含内部数据结构。

（3）如果数据对象的关键属性多于一个，那么其他的非关键属性必须表示整个数据对象而不是部分关键属性的特征。

例如，如果在“汽车”数据对象中增加“经销商”属性并将其与标识码一起作为关键属性，那么，再添加“经销商地址”属性就违背了上述规则，因“经销商地址”仅仅是“经销商”的特征，它与汽车的“标识码”无关。

（4）所有的非关键属性必须表示整个对象而不是部分属性的特征。

例如，在“汽车”数据对象中，增加“油漆名称”属性就违背了上述原则，因为它仅仅

与“颜色”有关，而不是整个“汽车”的特征。

### 5.2.2 实体—关系图

实体—关系图（Entity-Relationship Diagram）是制定产品规格说明书的一种图形语言机制。它是由美籍华人陈平山于 1976 年提出的。

通常，使用实体—关系图来建立数据模型。常把实体—关系图简称为 ER 图；相应地，用 ER 图描绘的数据模型也称为 ER 模型。

1. ER 图的基本成分

ER 图中包含了实体（即数据对象）、关系和属性三种基本成分。通常用矩形框代表实体，用连接相关实体的菱形框表示关系，用椭圆形或圆角矩形表示实体（或关系）的属性，并用无向边把实体（或关系）与其属性连接起来。

为了便于区分，ER 模型中的实体、关系和属性都应在对应的框中写上各自的名字。

2. 基数与模态

基数是指在一个给定的关系中实体间数量上的对应。基数通常简单地用“1”或者“多”来表示，它描述了“重复性”。

模态表示在一个关系中一个特定的实体是否必须参与的信息。

如图 5-4 所示，实体“教师”旁有两条竖线，靠近实体“教师”的竖线代表了“1 位教师”；另一条竖线代表了“必须”由教师来教课程。另一个实体“课程”旁有多分支线和圆圈，多分支线代表了“多”门课程，圆圈代表了“可以教也可以不教”课程。也就是说一个教师可以教多门课程，也可以不教课程；但是，课程必须由教师来教。

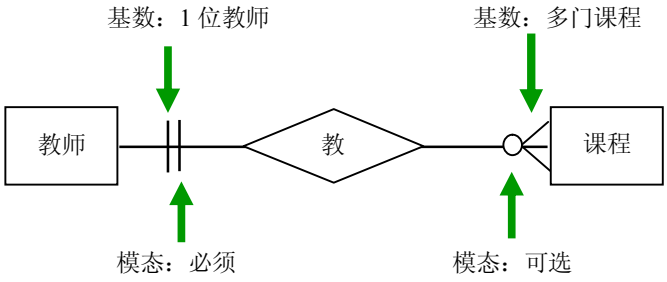



图 5-4 基数与模态

3. ER 图示例

如图 5-5 所示是汽车销售管理问题的 ER 图。其中，一个制造商生产一辆或多辆汽车，它可以给一个或多个经销商发放经销许可证。车辆可以通过一个或多个经销商销售，或者由厂家直接销售等。图 5-5 省略了部分实体或关系的属性。

人们通常用实体、联系和属性这三个概念来理解现实问题，因此 ER 模型比较接近人的习惯思维方式。此外，ER 模型使用简单的图形符号表达系统分析员对问题域的理解，不熟悉计算机技术的用户也能理解它。因此，ER 模型可以作为用户与分析员之间有效的交流工具。目前较为流行的 ER 图开发工具有 Power Designer 以及 ER-WIN 等。

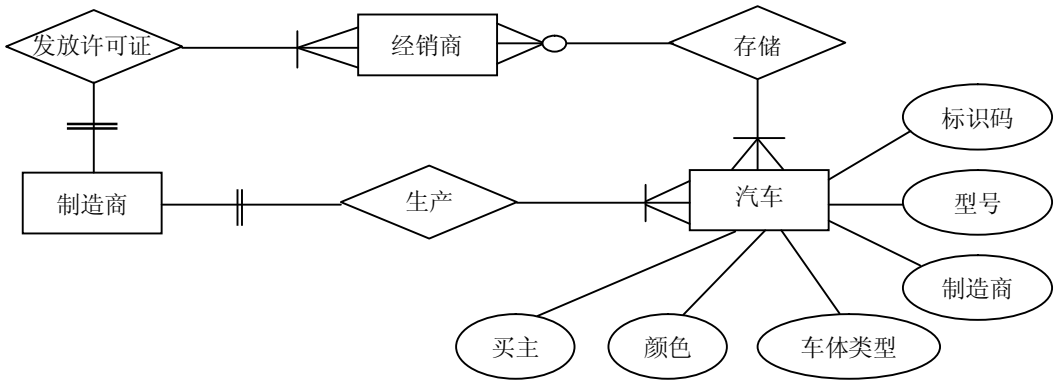



图 5-5　实体—关系图实例

## 5.3　基于数据流的分析方法

本节结合“家庭保安系统”讨论一些常用的启发式经验知识和规则，从而为分析人员建造易于理解的、描述用户需求的数据流模型提供方法上的指导。

### 5.3.1　创建数据流模型

数据流图是目标软件系统中各个子加工以及它们之间的数据流动的图形表示。数据流图的精化过程实际上是子加工和数据流的细化过程。随着这一过程的进行，用户需求逐步精确化、一致化和完备化。

在创建用户需求的数据流模型的过程中，分析人员应遵循以下规则：

（1）建立顶级数据流图，其中只含有一个代表目标软件系统整体处理功能的加工。

根据软件系统与外部环境的关系确定顶级数据流图中的外部实体以及它们与软件系统之间的数据流。

基于 4.2.1 节的初步需求分析结果，“家庭保安系统”的顶级数据流图如图 5-2 所示。

（2）对用户需求的文字描述进行语法分析，其中的名词和名词短语构成潜在的外部实体、数据源或数据流，动词构成潜在的加工。

结合分析人员对问题域和用户需求的理解，确定软件系统的主要功能以及它们之间的数据流。例如图 5-6，就是对图 5-2 的分析结果。

（3）采用通常的功能分解方法，按照“强内聚、松耦合”的原则逐个对加工进行精化；与此同时逐步完成对数据流的精化，并针对被精化的加工生成下一级数据流图。

“强内聚、松耦合”的原则是指被分解出来的各子加工之间的联系相对松散、简单，子加工内部各部分的联系相对紧密、复杂。这一原则对于目标软件系统的可修改性、可扩充性大有益处，因为开发人员可以缩小软件修改或扩充的影响传播范围。

对数据流的精化包含两个方面的意义。一方面，伴随着功能分解的进行，数据流的内容及各项特征将逐步彰显，所以要将其作为数据字典的一个条目，并不断精化、调整内容；另一方面，在父数据流图中的复合数据项可被分解为子数据项，这种数据流分解不能违背平衡原则。

例如，如果将图 5-6 中的“启动/停止处理”功能分解为“启动系统”和“停止系统”，那么“启动/停止命令”应相应地精化为“启动命令”和“停止命令”。

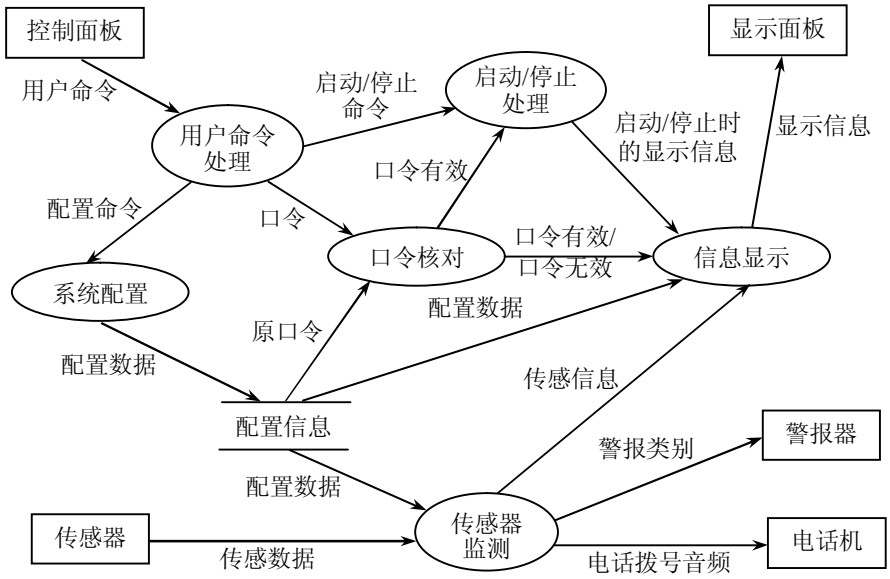



图 5-6 “家庭保安系统”1 级数据流图

对“家庭保安系统”中“传感器监测”子功能进行分解，得到了“家庭保安系统”的 2 级子数据流图如图 5-7 所示。

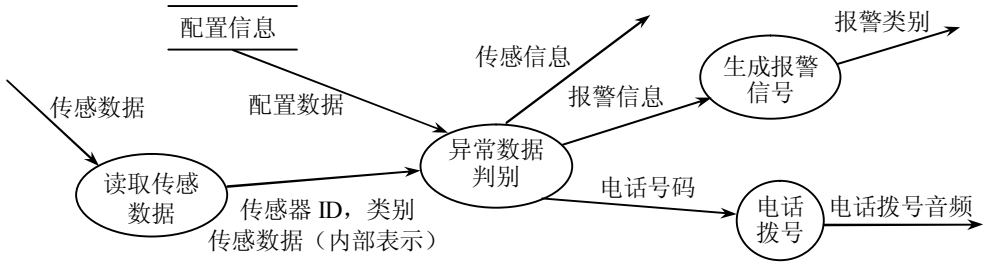



图 5-7 “家庭保安系统”2 级数据流图：对“传感器监测”的分解

（4）精化过程中必须维持各级数据流图之间的数据流平衡。

（5）精化过程应适可而止，避免涉及软件设计细节。一般说来，如果某子功能可以用一段简洁、精确的文字描述清楚，就无须进一步分解。

### 5.3.2 过程规格说明

对于数据流图中不再分解的加工，分析人员要借助结构化自然语言对其功能进行精确、简洁的描述。

图 5-6 中“口令核对”子功能分解出来的“设置口令”子功能可描述如下：

（1）参数：口令；类别：字符串。

（2）处理步骤如下：

①检查系统是否已有口令。若有，则验证用户输入口令的有效性。如果有效，则显示提示信息要求输入新口令；否则，显示失败信息并退出。

②检查口令长度是否合法。如果非法，则显示提示信息要求重新输入。

③要求用户再次键入合法口令，以便用户确认和记忆。如果两次键入的口令不符，则返回。

④将确认后的口令按某种加密方法转换为另一字符串存放于系统配置文件中。显示成功信息并退出。

（3）约束条件：

在上述①、②、③步骤中，用户重试的机会不超过 3 次。

## 5.4　基于 CASE 工具的需求分析

使用前述方法进行需求分析时，需要计算机在数据流图的绘制、数据字典的存储、检索及一致性检查等方面提供帮助。因此，本节给出一个基于数据流图的需求分析 CASE 工具的蓝本 DFA_Tool。考虑到传统的数据流图语言机制和分析方法对大中型软件开发的支持能力比较单薄，我们结合结构化分析 CASE 工具的发展趋势，在需求分析的方法学、语言机制和分析能力等方面进行了扩充，引进了具体的多视点分析方法、可视形式化和可执行的需求描述语言以及动态分析技术。

以下分别介绍：DFA_Tool 的主要思想；具有可视形式化特征的语言机制；利用 DFA_Tool 进行需求分析的方法。

### 5.4.1　核心思想

CASE 工具的开发必须基于某些基本的原则或思想。这些原则或思想不仅决定着 CASE 工具的开发过程，而且也将对基于该 CASE 工具的软件开发行为发挥重要的指导作用。DFA_Tool 的核心思想可以归纳为：多视点需求分析、可视形式化和可执行的需求规格说明语言。

1. 多视点需求分析

DFA_Tool 从相互关联的结构、功能和行为三个方面分别为目标软件系统建立模型。

在结构视点，分析人员可根据系统的物理结构或软件结构（例如，物理构件、软件模块、任务等）进行层次分解，并标识系统各部分之间的数据流，进而建立系统的结构图。

在功能视点，分析人员利用功能分解方法刻画系统的活动（类似于数据流图中的加工）以及活动之间可能出现的数据流，以逐层精化的方式建立系统的活动图。与数据流图一样，DFA_Tool 的活动图不包括任何动态性质。它既不关心活动是如何启动和终止的，也不关心活动能否与其他活动并行执行。至于数据流，活动图只说明它们可以在某些活动之间流动，并不指明何时流动。

应用系统在时间坐标系中的所有控制行为均由行为视点描述。对于层次结构中的每一级活动图，均有一个相应的行为图，它们刻画系统的动态行为，包括：

（1）由于各时间点上事件的刺激，某些活动被启动或终止，并引发新的事件。

（2）对活动的活跃情况及数据的流动情况进行连续监测，据此决定系统的下一步行为。

由此可见，系统的活动图（功能视点）和行为图（行为视点）是紧密耦合的，它们共同

构成系统的概念模型。结构图与活动图之间的关系是简单而直接的：结构图中的某些构件负责实现活动图中的某些功能。如图 5-8 所示是 DFA_Tool 的模型结构。

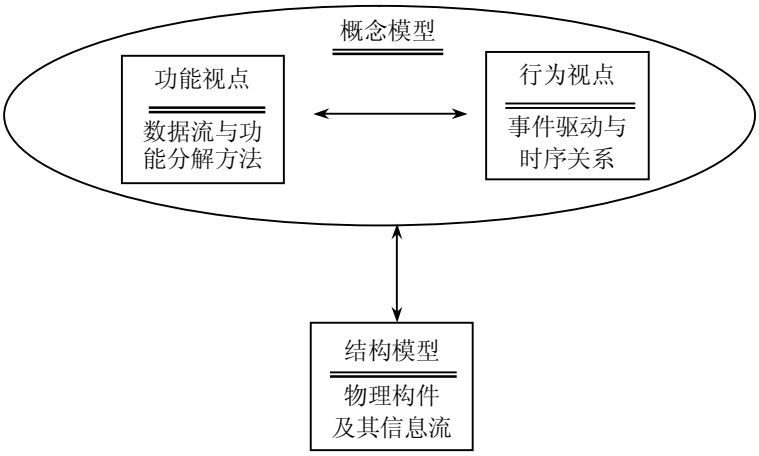



图 5-8 DFA_Tool 的模型结构

DFA_Tool 可以自动完成三个视点之间的某些一致性检查。例如，结构图中的数据流是否与活动图的数据流一致，结构图中哪些模块未产生输出，活动图中哪些活动永远不会被启动。

2. 可视形式化

与文字相比，图形更为直观、简洁。因此，需求分析活动往往离不开图形机制的支持，例如前述的数据流图。DFA_Tool 的结构图、活动图及状态图均基于一组共同的简单图形记号，并且都具有形式化语义。因此，DFA_Tool 兼备了图形的简单直观和形式化机制的精确严谨。

3. 可执行的需求规格说明语言

对于大中型软件项目的开发，未对需求规格说明书进行详细的测试便进行软件设计往往是危险的。因此，用于表述需求规格说明书的语言机制应该具有可执行语义，以便系统能够在需求分析阶段实际展示目标软件系统的动态行为，为分析人员提供动态分析、调试和测试等手段，并让用户尽早进行需求确认。基于以上考虑，DFA_Tool 为图形语言机制定义了操作语义，并且它不仅可以用步进方式，还可以用预编程方式演示系统的动态行为。DFA_Tool 提供一种元级模拟控制语言（Simulation Control Language，SCL）。借助 SCL，分析人员可以采用各种方式模拟外部环境，捕获目标软件系统的状态或事件，并进行元级控制，例如模拟执行至某个预设断点，跳过不重要的中间状态或者忽略一些琐碎的执行细节。

### 5.4.2 基于 CASE 工具的需求分析

基于 CASE 工具的需求分析方法大体上仍与 5.3 节类似。但是，针对具体的 CASE 工具，有必要对传统的方法和步骤做局部调整，以适应该 CASE 工具所确立的需求分析方法学。例如，基于 DFA_Tool 的需求分析过程大致如下：

（1）从多个视点分别建立目标软件系统的结构模型、功能模型和行为模型。

（2）在上述建模过程中可以采取功能分解、逐步求精的方法对部分子系统展开分析活动。一般而言，首先应建立子系统的活动图（功能模型），然后再给出相应的状态图（行为模型）。

（3）利用 DFA_Tool 表格说明机制对各种图形表示中的所有图元的内容进行描述，这些图元包括数据流、事件、状态和原子活动等。

（4）利用 DFA_Tool 的动态分析能力对目标软件系统或其子部分的模型进行模拟执行，以便发现不一致性和不完全性，并进行相应的修改完善。

DFA_Tool 可以为分析人员自动完成以下烦琐任务：

（1）模型的图示、存储与检索。

（2）模型之间、数据条目之间的一致性检查。

（3）行为模型中状态的可达性分析、死锁分析。

（4）模型的动态模拟执行。

（5）模型修改的影响传播范围的确定等。

## 习题

1．简述数据流图的主要思想，概述使用数据流图进行需求分析的过程。

2．可以分别采用数据流方法中的哪些技术来完成用户需求的精确化、一致化和完全化任务？

3．实体—关系图所适用的领域的主要特征有哪些？

4．如何判断数据流图的一致性和完全性？

5．在数据流图中，可否将两个加工用一个数据流相连？可否将两个数据源用一个数据流连接？为什么？

6．针对第 4 章习题 5 所述的图书馆管理问题，采用实体—关系图表示该问题所涉及的主要实体、属性及实体之间的相互关系。

7．选取一个你比较熟悉的中小型软件问题，分别采用或综合采用数据流图、实体—关系图进行部分需求分析工作，要求：

（1）至少给出第 0～2 级数据流图。

（2）给出相应的数据字典和实体—关系图。

8．简述 5.4 节所述的需求分析 CASE 工具 DFA_Tool 的主要思想。

9．基于第 4 章习题 5 所给出的初步需求文档，仍以原来的分组为单位使用数据流方法完成需求建模任务。要求给出以数据流图和数据字典为主体的需求规格说明书，然后进行严格的需求评审。