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。

抽象和分解是处理任何复杂问题的两个基本手段。

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。

结构化分析将数据和处理(加工)作为分析对象，数据的分析结果表示了现实世界中实体的属性及其之间的相互关系，而处理的分析结果则展现了系统对数据的加工和转换。

**数据流图**也称数据流程图(Data Flow Diagram，DFD),它是一种便于用户理解、分析系统数据流程的图形工具。它摆脱了系统的物理内容，精确地在逻辑上描述系统的功能、输入、输出和数据存储等，是系统逻辑模型的重要组成部分。



DFD (数据流图)将系统建模成输入—处理—输出的模型，即流入软件的数据对象，经由处理的转换，最后以结果数据对象的形式流出软件。DFD使用分层的方式表示，第一个数据流模型有时被称为第0层DFD或者环境数据流图。从整体上表现系统，随后的数据流图将改进第0层图，并增加细节而信息。

除DFD外，再进行建模时，还可结合**数据字典**和**加工处理说明**对DFD进行补充。

数据字典以一种准确且无二义的方式定义所有被加工引用的数据流和数据存储，通常包括**数据流条目**、**数据存储条目**和**数据项条目**。

数据流条目描述DFD中数据流的组成；

数据存储条目描述DFD中数据存储文件的组成；

数据项条目则描述数据流或数据存储中所使用的数据项。

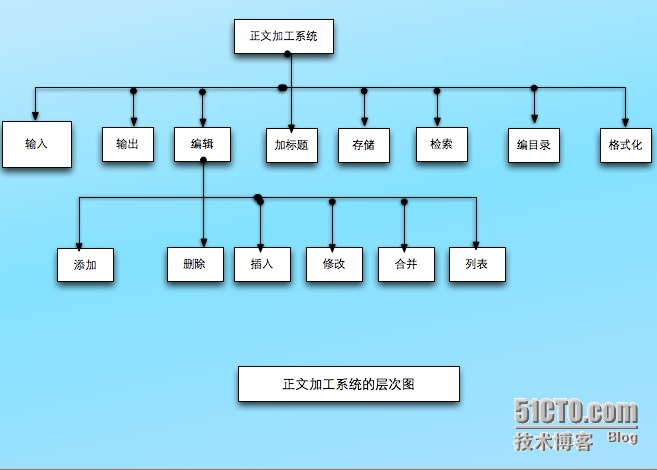
加工处理的说明则可采用结构化自然语言、判定表和判定树等多种形式进行详细描述，其目的在于说明加工做什么。

掌握上述的工具后，即可对问题进行结构化的分析，其实施步骤如下：

1. 确定系统边界，画出系统环境图；
2. 自顶向下画出各层数据流图；
3. 定义数据字典；
4. 定义加工说明；
5. 将图、字典以及加工组成分析模型。

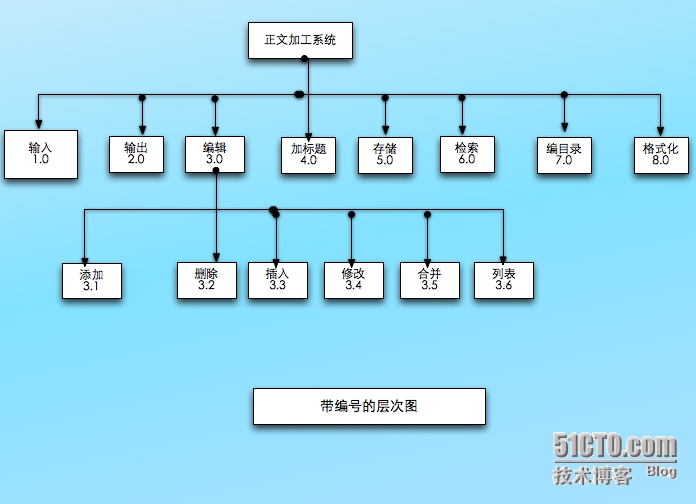
DFD、数据字典和加工说明可以充分地描述系统的分析模型，其后需要对分析模型进行变换从而得到系统的总体设计模型。系统总结设计模型可以采用层次图、HIPO图和结构图来表达。

层次图用来描述软件的层次结构。虽然层次图的形式和描绘数据结构的层次方框图相同，但是表现的内容却完全不同。层次图中的一个矩形框代表一个模块，方框间的连线表示调用关系而不像层次方框图那样表示组成关系。下图是层次图的一个例子，最顶层的方框代表正文加工系统的主要模块，它调用下层模块完成正文加工的全部功能；第二层的每个模块控制完成正文加工的一个主要功能，例如“编辑”模块通过调用它的下属模块可以完成6钟编辑功能中的任何一种。

[](http://s3.51cto.com/wyfs02/M01/24/88/wKioL1NRHtzw21TNAAIxA3hyv8w381.jpg)

层次图很适合在自顶向下设计软件的过程中使用。

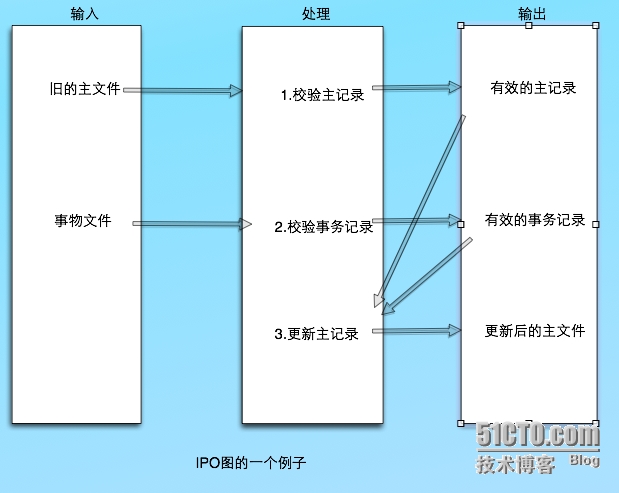
HIPO图是美国IBM公司发明的“层次图加输入/处理/输出图”的英文缩写。为了能使HIPO图具有可追踪性，在H图（层次图）里除了最顶层的方框之外，每个方框都加了编号。如下图：

[](http://s3.51cto.com/wyfs02/M00/24/88/wKioL1NRHv_B2-AKAAJ55I058-o217.jpg)

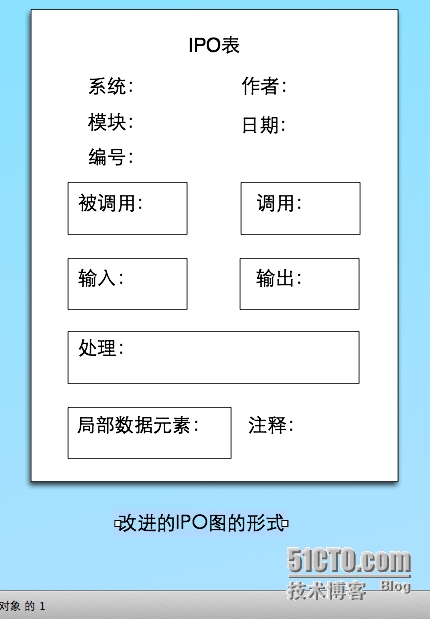
和H图中每个方框相对应，应该有一张IPO图描绘这个方框代表的模块的处理处理过程。下面再详细介绍IPO图。但是有一点应该指出，那就是ＨＩＰＯ图中的每张IPO图内都应该明显地标出它所描述的模块在H图中的编号，以便追踪了解这个模块在软件结构中的位置。

IPO图是输入、处理、输出图的简称，它是由美国IBM公司发展完善起来的一种图形工具，能够方便地描绘输入数据，对数据处理和输出数据之间的关系。

IPO图使用的基本符号既少又简单，因此很容易学会使用这种图形工具。它的基本形式是在左边的框中列出有关的输入数据，在中间的方框内列出主要的处理，在右边的框内列出产生的输出数据。处理框内列出处理的次序暗示了执行的顺序，但是用这些基本符号还不足以精确描述执行处理的详细情况。在IPO图中还用类似向量符号的粗大箭头清楚的指出数据通信的情况。下图是一个主文件更新的例子，通过这个例子不难了解IPO图的用法。

[](http://s3.51cto.com/wyfs02/M02/24/88/wKiom1NRH1uy5jiVAAGdE-U3tHM848.jpg)

小编建议使用一种改进的IPO图（也成为IPO表），使用图中包含某些附加的信息，在软件设计过程中将比原始的IPO图更有用。下图是改进的IPO图中包含的附加信息主要有系统名称、图的作者、完成的日期、本图描述的模块的名字、模块在层次图中的编号、调用本模块的模块清单、本模块调用的模块的清单、注释以及本模块使用的局部数据元素等。在需求分析阶段可以使用IPO图简略地描述系统的主要算法（即数据流图中各个处理的基本算法）。

[](http://s3.51cto.com/wyfs02/M01/24/88/wKioL1NRH0mj8OT0AAFYttmFcqo150.jpg)

当然，在需求分析阶段，IPO图中的许多附加信息暂时还不具备，但是在软件设计阶段可以进一步不充修正这些图，作为设计阶段的文档。这正是在需求分析阶段用IPO图作为描述算法的工具的重要优点。

在分析模型的基础上进行设计时，主要是针对DFD进行变换从而得到模块的调用关系图，因此，需要掌握数据流的变换设计与事务设计。

面向数据流的设计方法把数据流图映射成软件结构，数据流图的类型决定了映射的方法，数据流图可分为变换型数据流图和事务型数据流图。

**变换型数据流图**具有明显的输入、变换（或称主加工）和输出。

**事务型数据流图**是数据沿输入通路到达一个处理时。这个处理根据输入数据的类型在若干动作序列中选择一个来执行。

变换设计的核心在于确定输入流和输出流的边界，从而孤立出变换中心；

事务设计的核心在于将事务类型判断处理变换成调度模块以选择后续的输出分支模块。

经过总体设计阶段的工作，已经确定了软件的模块结构和接口描述，但每个模块仍被看作黑盒子。后续的详细设计目标是确定怎样具体地实现所要求的系统，经过详细设计，可以得出对目标系统的精确描述，从而在编码阶段可以将这个描述直接翻译成用某种程序设计语言书写的程序。因此，详细设计的结果基本上决定了最终的程序代码的质量。详细设计可以采用程序流程图、N-S图、PAD图和PDL语言等工具来表达。