

## 第4章 结构化设计方法

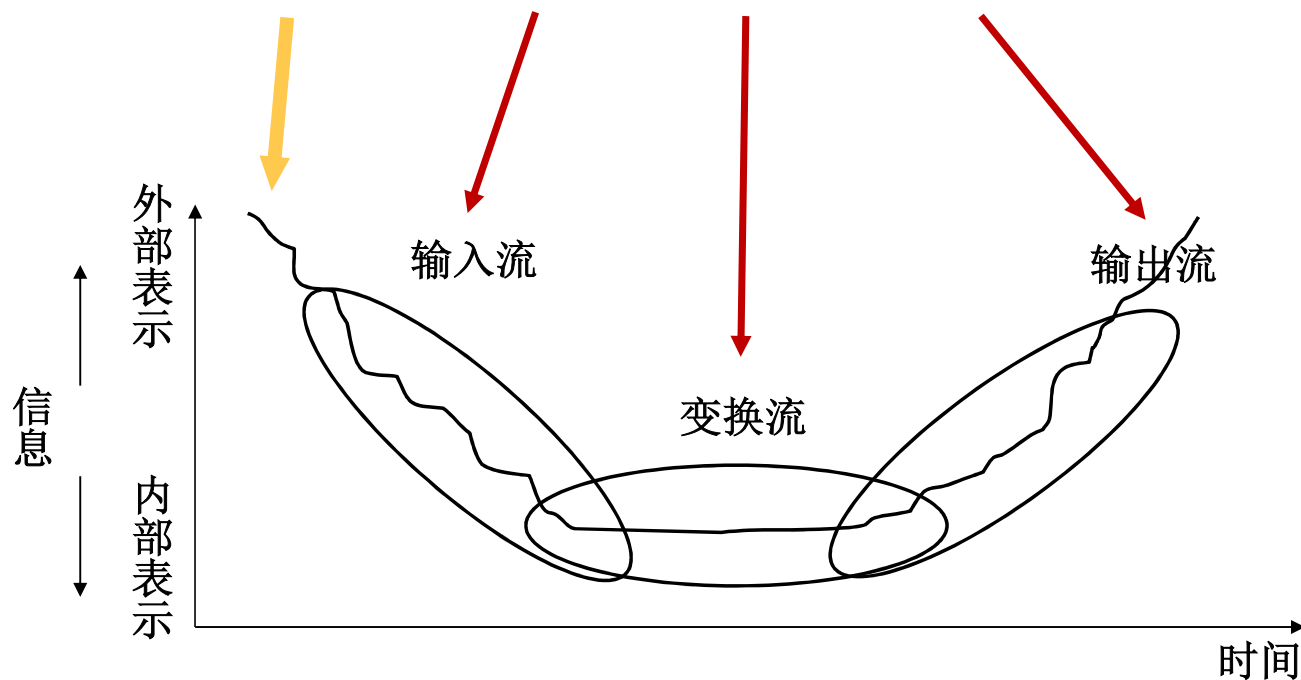
---

- ❖ 结构化设计方法概述
- ❖ 面向数据流的设计方法
- ❖ 结构化详细设计的工具

# 结构化设计方法概述

## 面向数据流的设计方法 —— 变换分析法

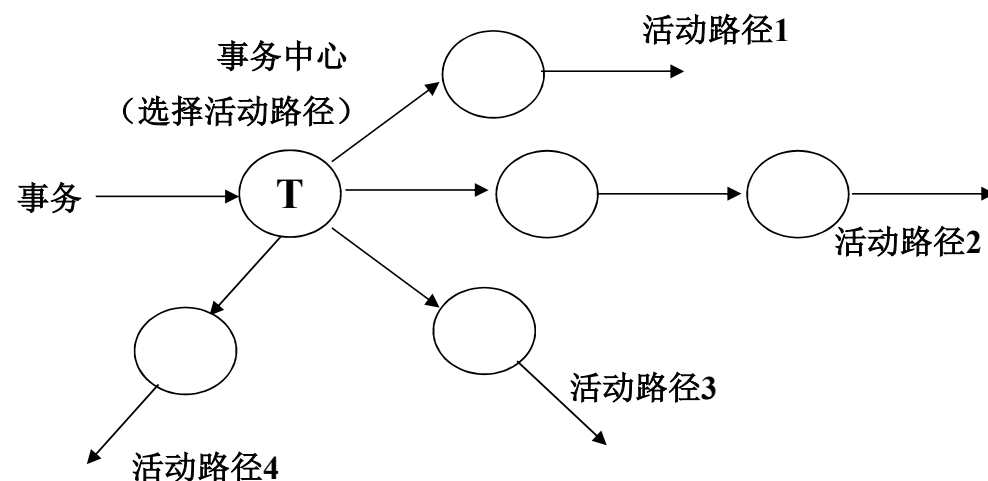
在数据流图中，系统数据流通过输入流、变换流、输出流，完成对数据的分析处理。



# 结构化设计方法概述

## 面向数据流的设计方法 —— 事务分析法

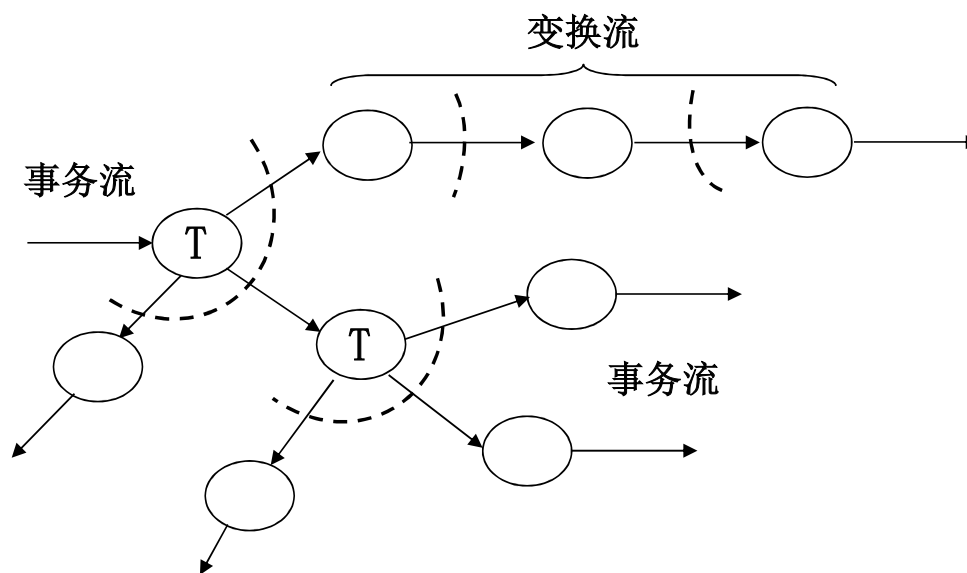
事务分析法是特殊的变换分析法，它的典型特征在于数据流图中有一个“事务中心”，它处理从多条变换输出路径中选择一条活动路径，具有这种选择处理能力的加工逻辑（模块）就称为事务中心。



# 结构化设计方法概述

## 面向数据流的设计方法 —— 混合分析法

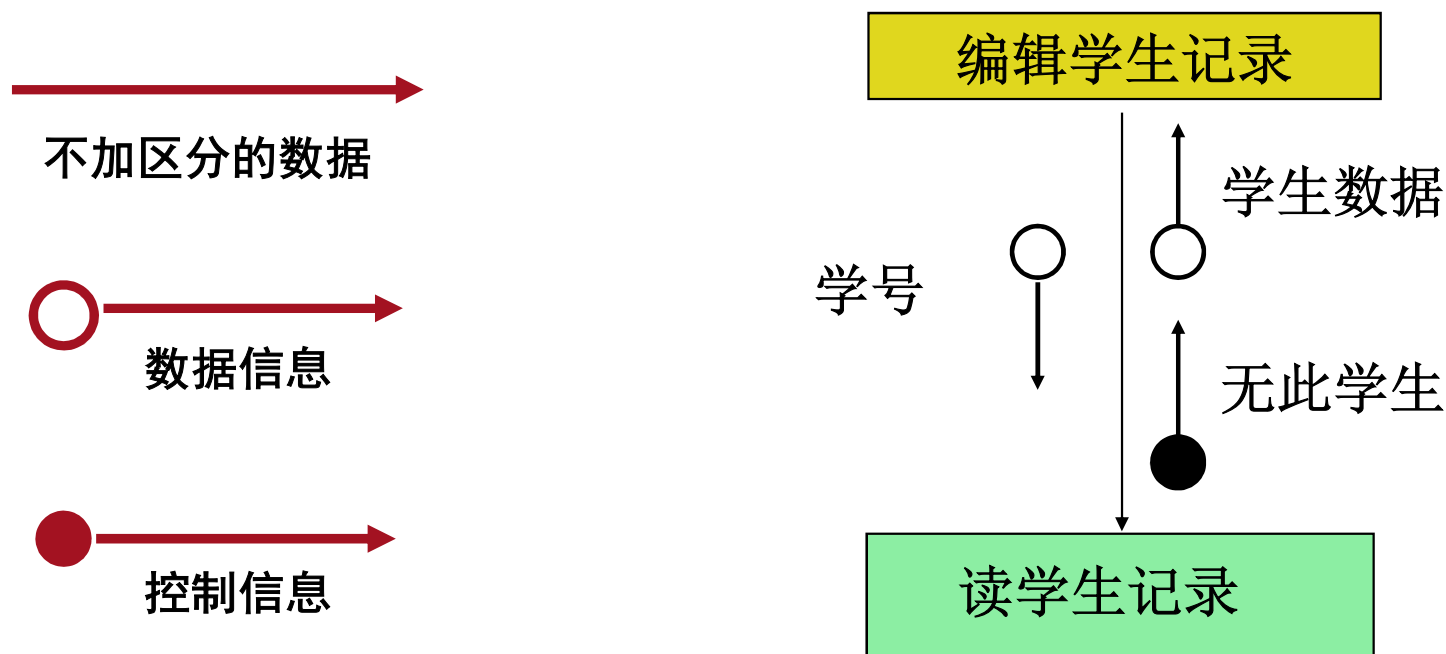
在一个大型系统的数据流图中，**变换流**和**事务流**会同时出现。按照结构化设计中分解的思想，上层数据流图整体反映一个主题：变换流或事务中心。在对数据流图分解的下层图或各条路径活动中，再确定变换流或事务中心。如此往复迭代，形成混合分析法。



# 面向数据流的设计方法

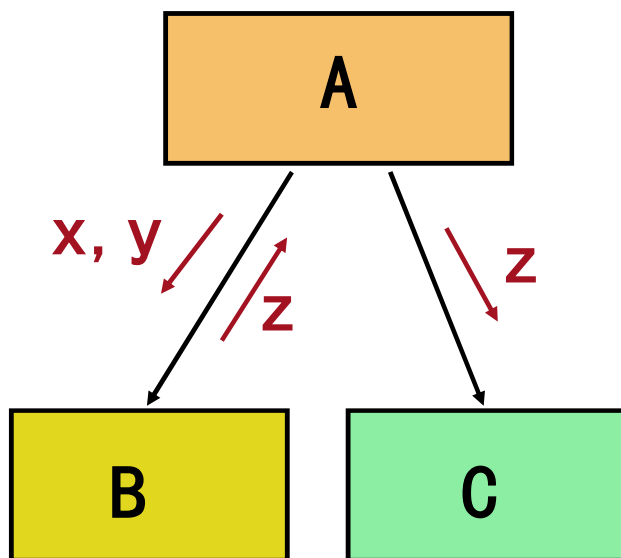
## 表示软件系统结构的图形工具：结构图

结构图用于软件系统结构的设计，以方框表示模块，方框间的连线表示调用关系。

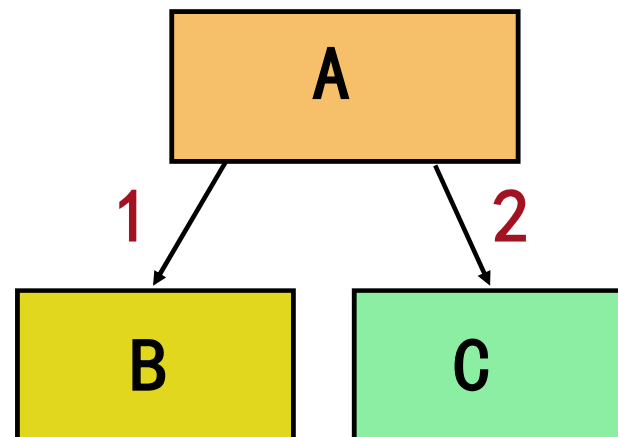


# 面向数据流的设计方法

## 1. 结构图——调用及数据流



(a)

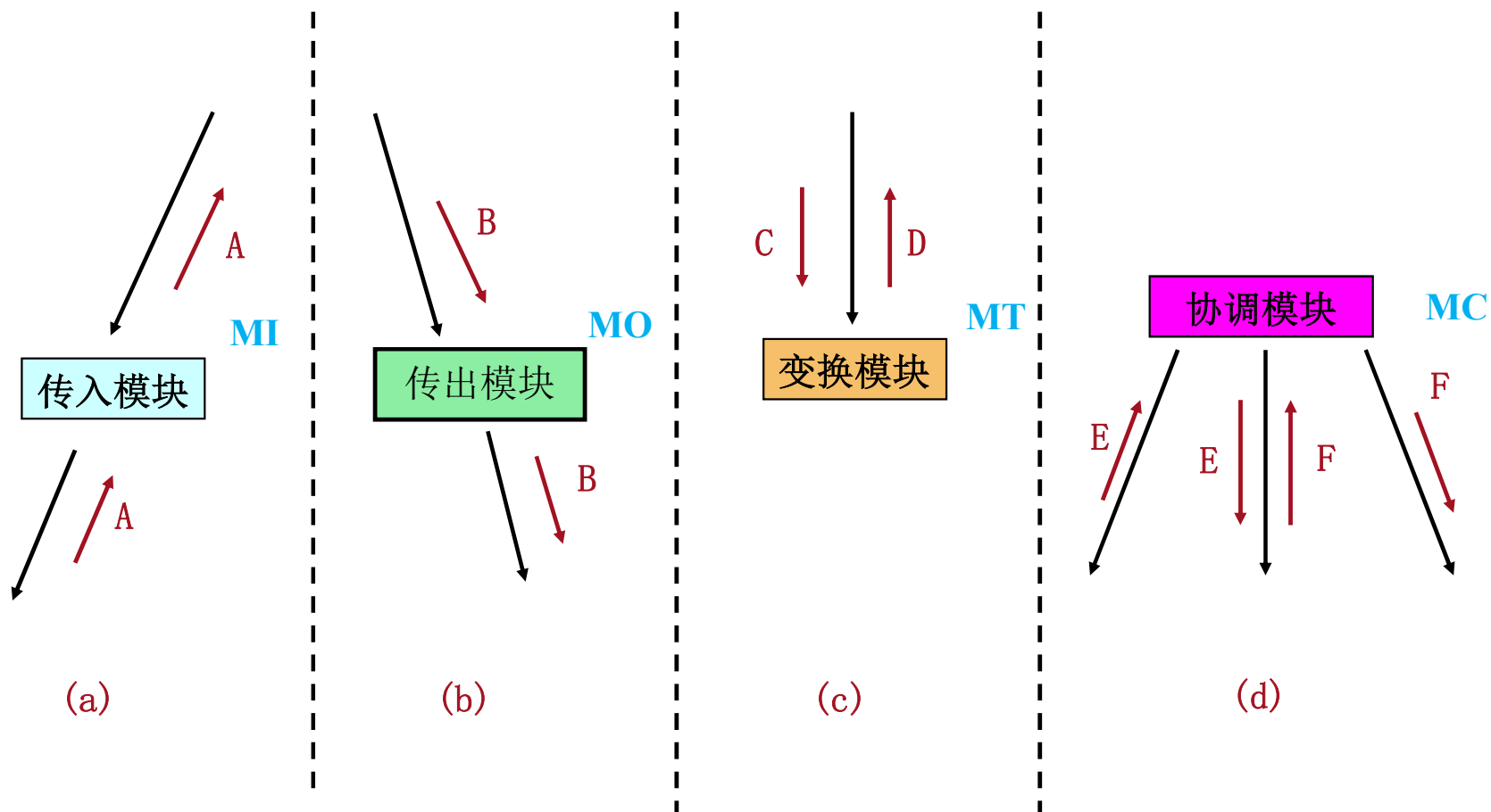


	入	出
1	x, y	z
2	z	—

(b)


# 面向数据流的设计方法

## 2. 结构图——四种基本模块调用方式



# 面向数据流的设计方法

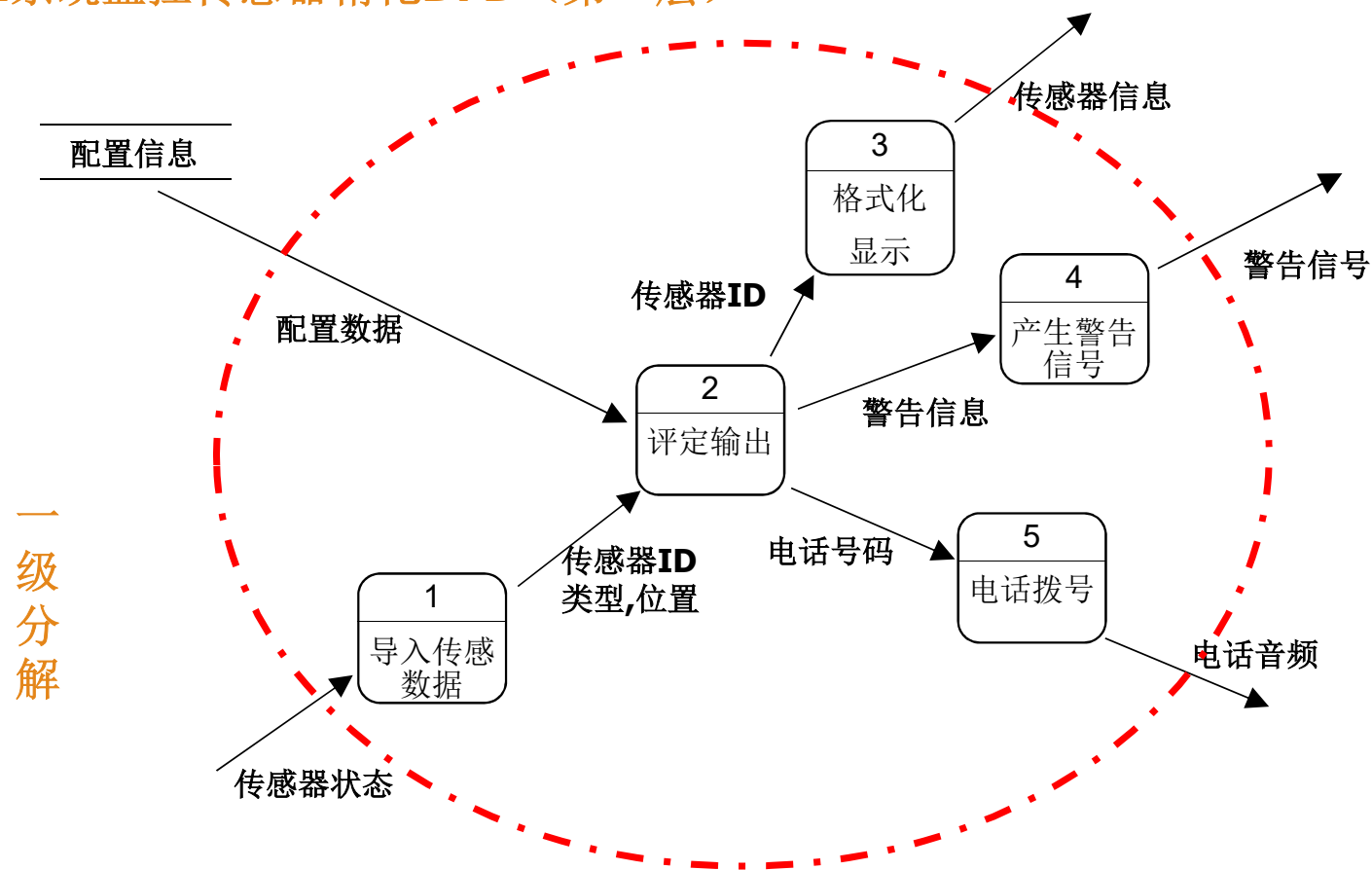
## 面向数据流设计方法的设计步骤——变换分析法

- (1) 确定数据流的类型，精化DFD；
  - 复审顶层数据流图
  - 分解数据流图
- (2) 划定数据流自动化边界，确定数据流类型；
- (3) 把DFD映射到系统模块结构，设计出模块结构的上层；
- (4) 基于分层的DFD，设计对应层次的模块结构；
- (5) 根据模块独立性原理，精化模块结构；



# 面向数据流的设计方法

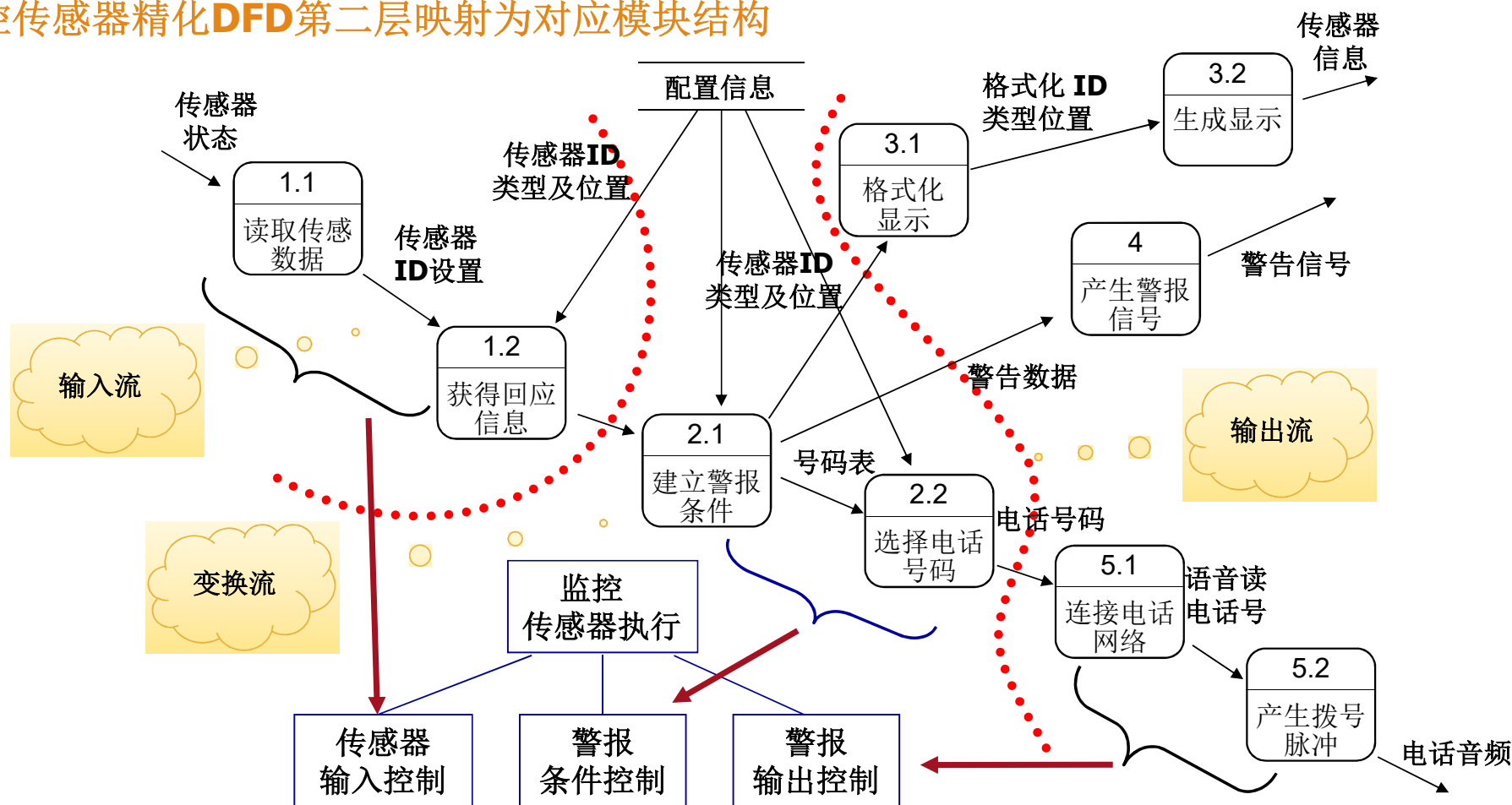
## SAFEHOME系统监控传感器精化DFD（第一层）



# 面向数据流的设计方法

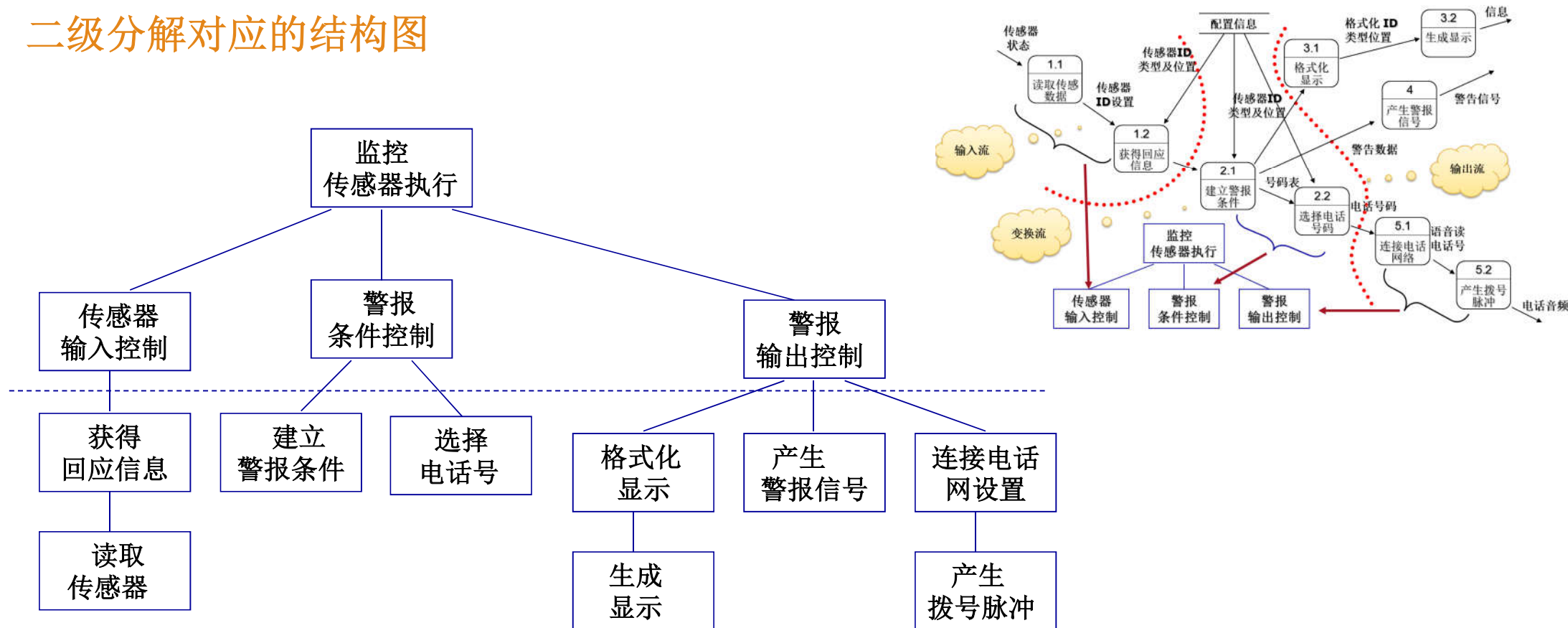
监控传感器精化DFD第二层映射为对应模块结构

二级分解

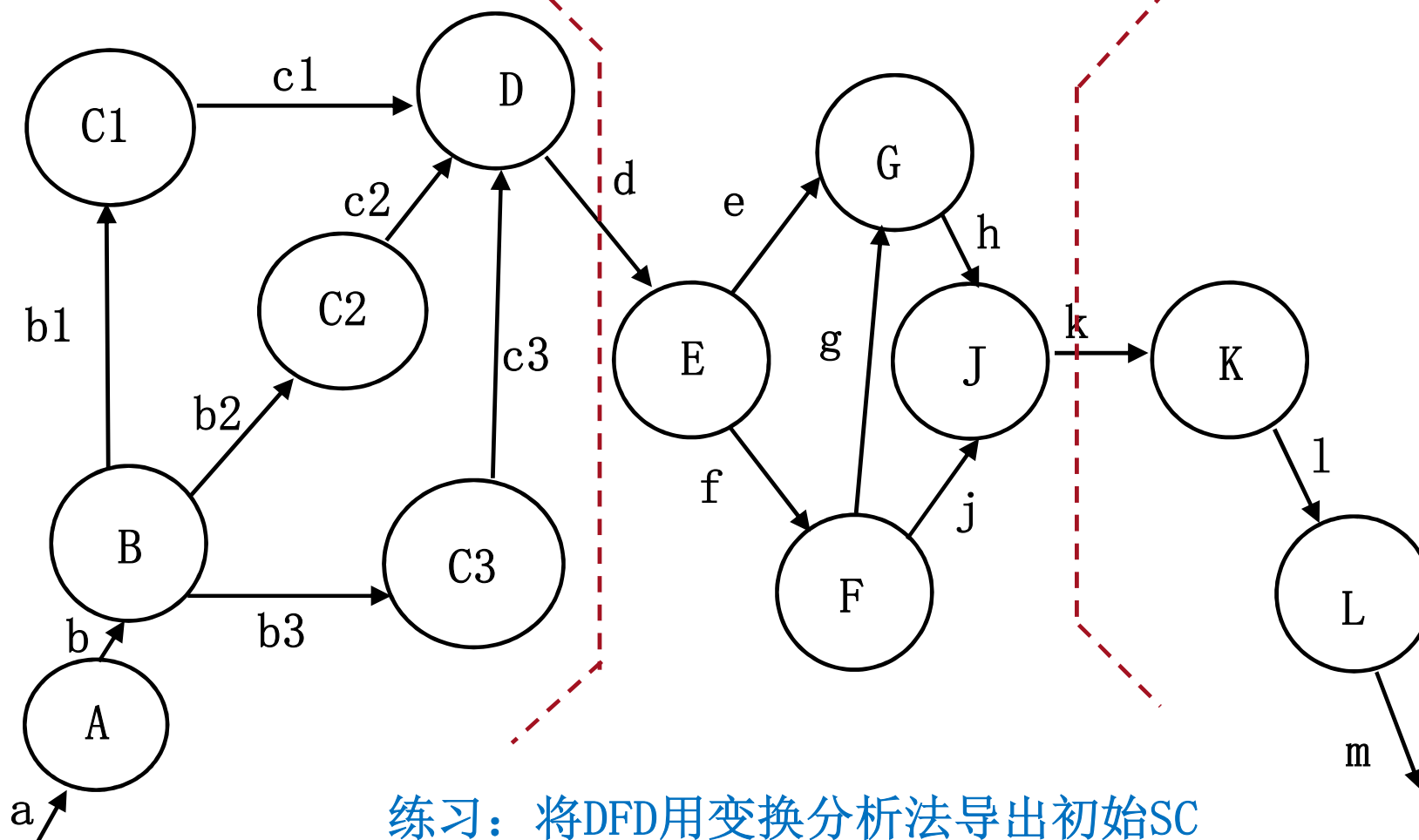


# 面向数据流的设计方法

二级分解对应的结构图



# 面向数据流的设计方法



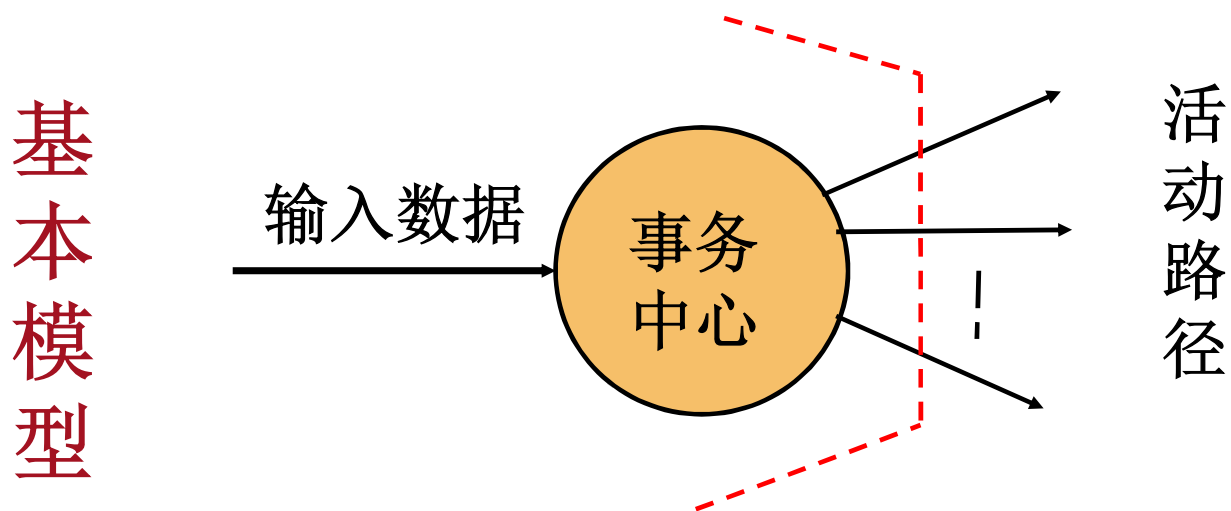
练习：将DFD用变换分析法导出初始SC

# 面向数据流的设计方法

## 事务分析法

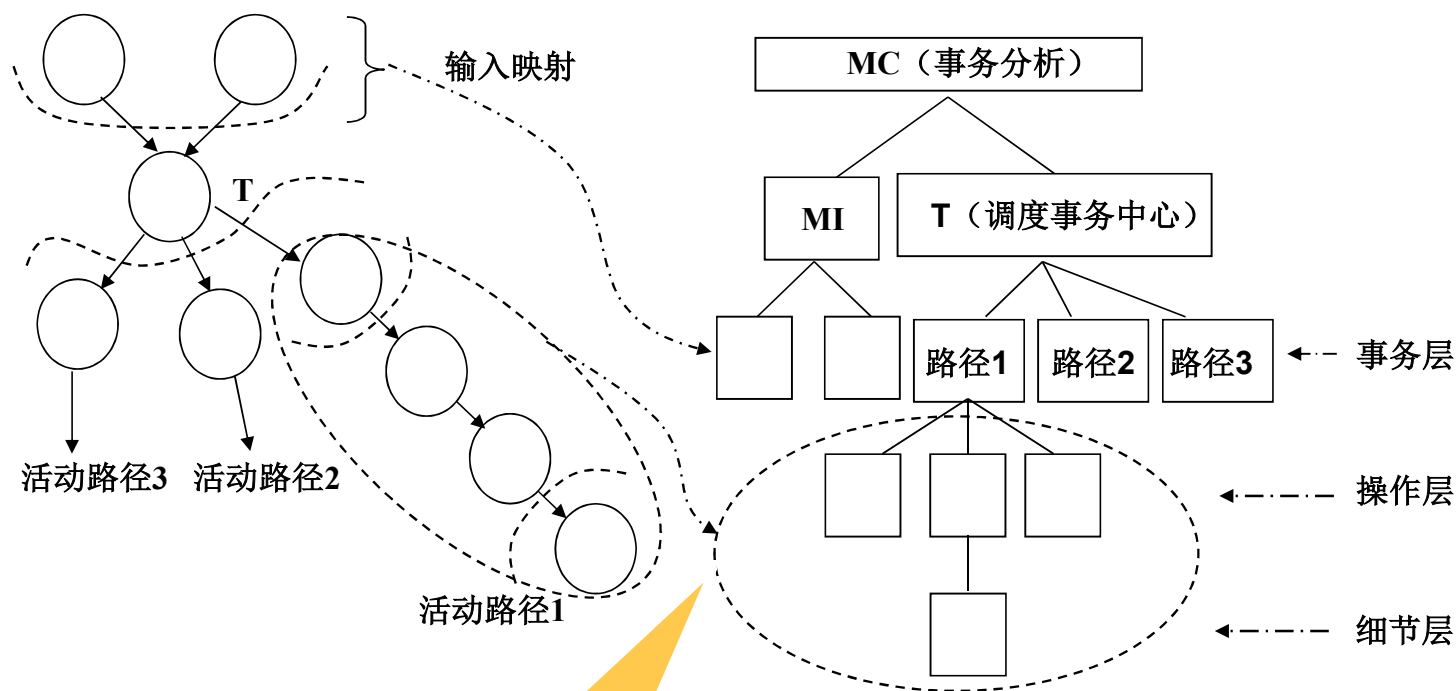
原则上，任何系统都可基于变换分析方法设计软件结构。但是，如果数据流具有明显的事务特征时，以采用事务分析方法为宜。

数据流图中的事务特征，是指能找到事务中心和对应的多条活动路径。



# 面向数据流的设计方法

## 活动路径分支的典型结构



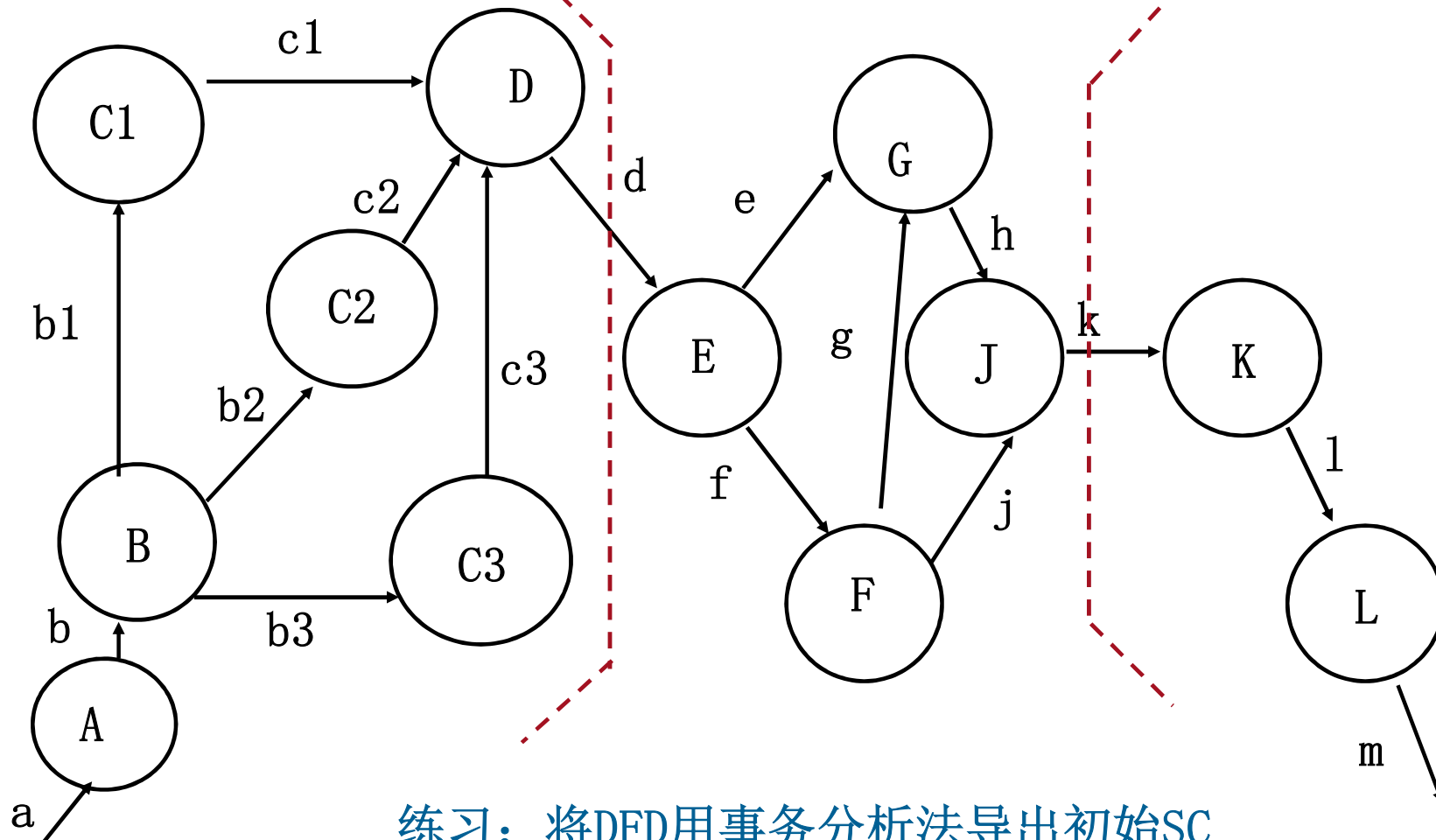
同时体现了：混合分析法

# 面向数据流的设计方法

事务分析设计方法步骤：

- (1) 复审并精华数据流图；
- (2) 确定数据流图特征，判断是变换流还是事务流；
- (3) 设定自动化边界，分离出事务中心和事务路径；
- (4) 执行“一级分解”；
- (5) 执行“二级分解”；
- (6) 采用启发式规则。

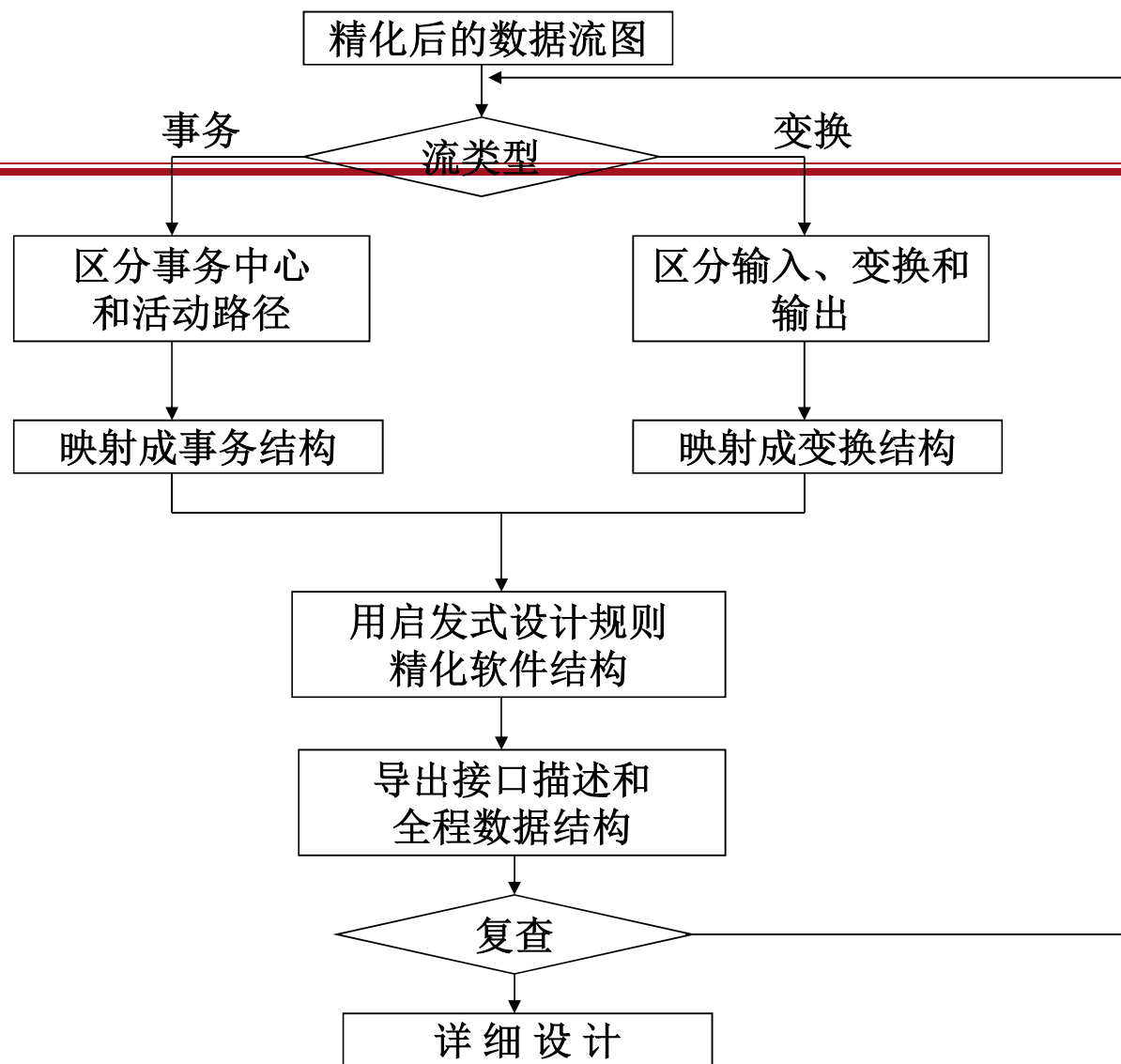
# 面向数据流的设计方法



练习：将DFD用事务分析法导出初始SC



## 面向数据流的设计过程



# 结构化详细设计的工具

结构化设计的详细设计阶段，主要完成系统各模块功能的**过程描述**。

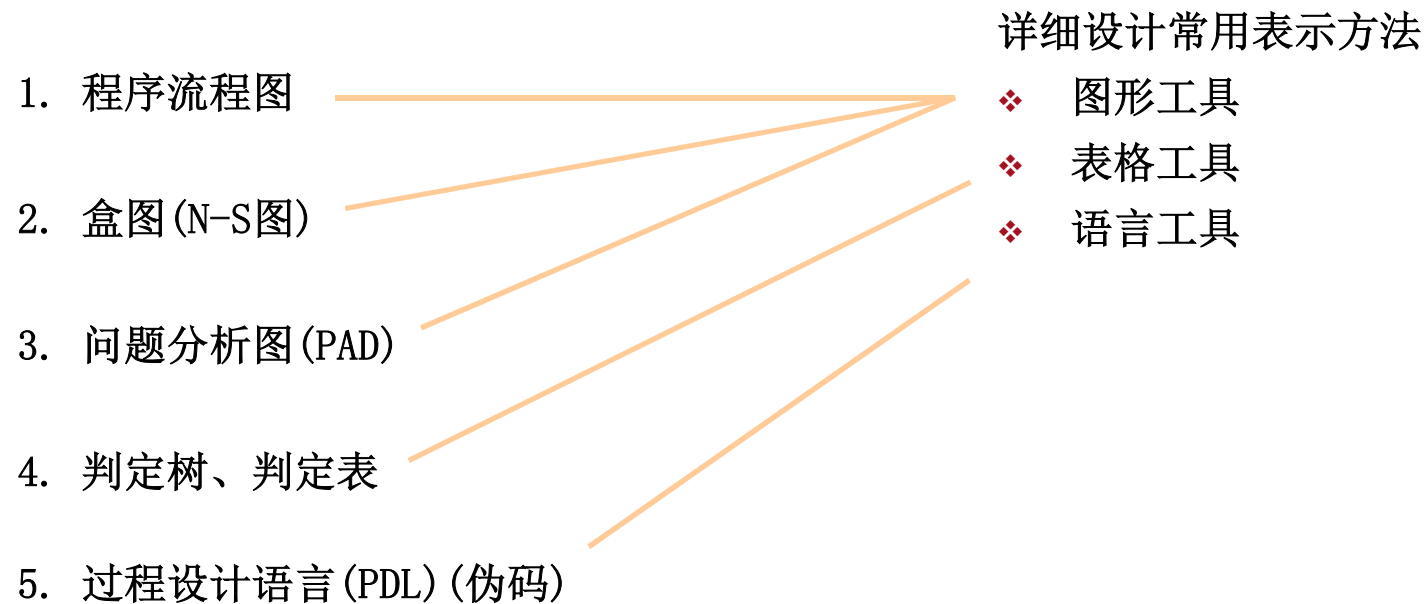


详细设计提供了**图形**、**表格**和**语言**等三类不同工具。

无论何种图形工具，都提供结构化程序设计对应的控制流程，以及功能的处理、数据的组织、数据结构的描述，以利于从详细设计到程序的实现。

# 结构化详细设计的工具

## 详细设计中常用的图形工具



# 结构化详细设计的工具

## 1. 程序流程图 (PFD—Program Flow Diagram)

控制结构

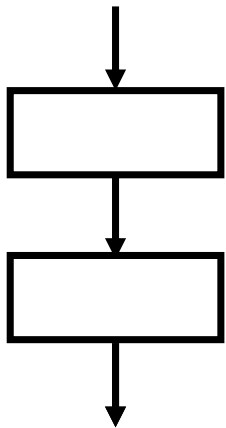
1. 顺序型 (图1)

2. 选择型 (图2)

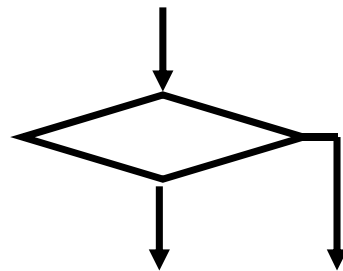
3. 循环型

先判断后循环 (图3.1)

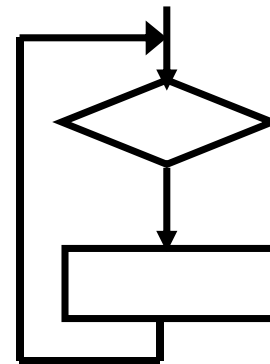
先循环后判断 (图3.2)



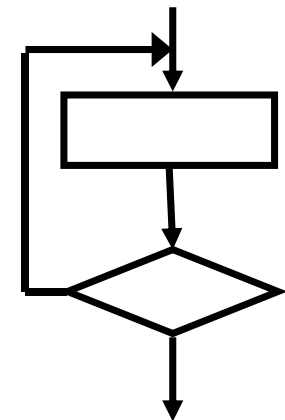
(图1)



(图2)



(图3.1)



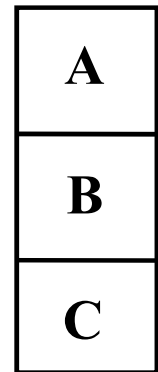
(图3.2)

# 结构化详细设计的工具

## 2 盒图(N-S图)

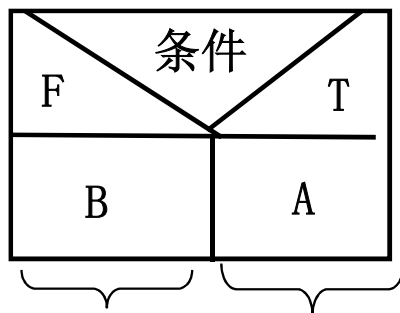
描述基本控制结构的图形构件

(1) 顺序型 

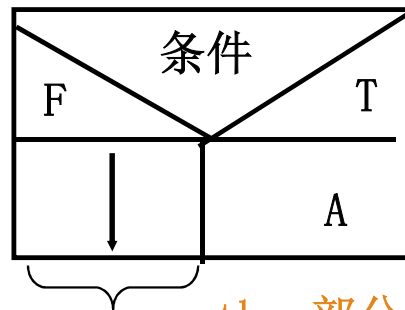


(2) 选择型

if - then - else      if - then

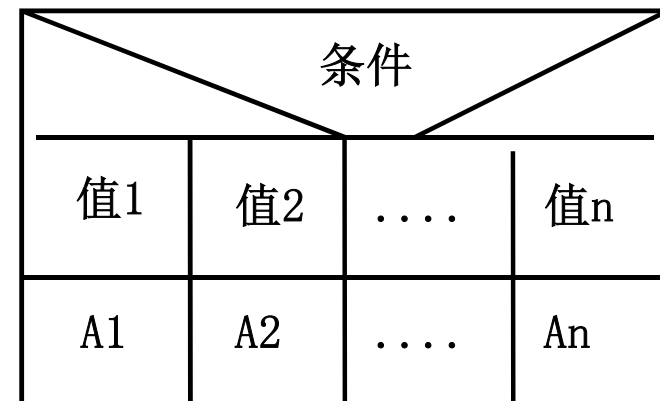


else部分    then部分



then部分

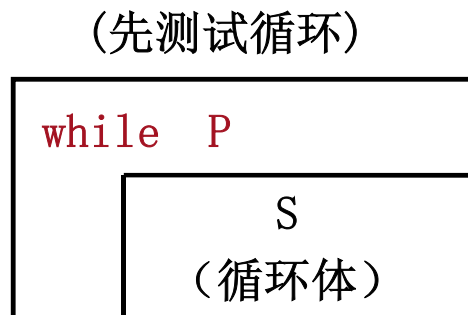
(3) 多分支选择型 (switch case)



# 结构化详细设计的工具

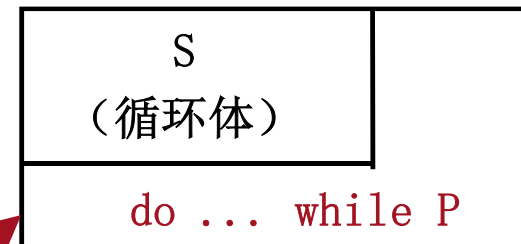
## 2 盒图(N-S图)

(4) while 循环



(5) do ... while循环

(后测试循环)



循环条件

# 结构化详细设计的工具

## 猜测随机数游戏 —— 盒图设计

盒图绘出实现过程：

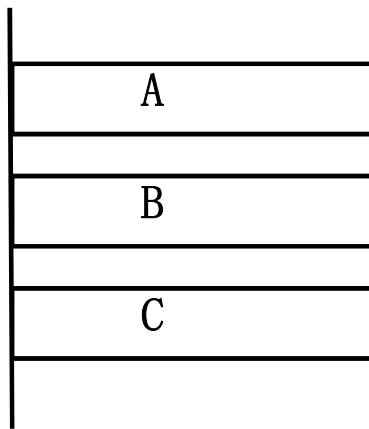
游戏生成一个随机数，之后用户输入一个整数。通过程序的提示信息：“大”、“小”，引导游戏者最终猜出系统生成的随机数，并显示游戏者猜测随机数的次数。

# 结构化详细设计的工具

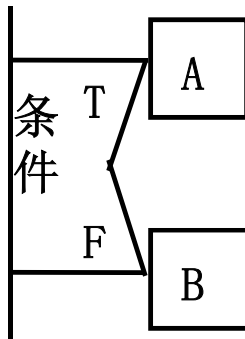
## 3 问题分析图(PAD) (Problem Analysis Diagram)

### (3) 重复结构

### 基本控制结构:

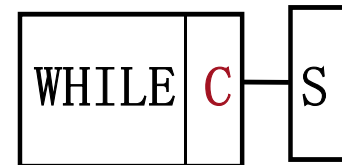


(1) 顺序结构

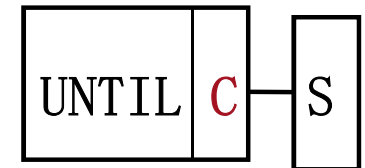


(2) 选择结构

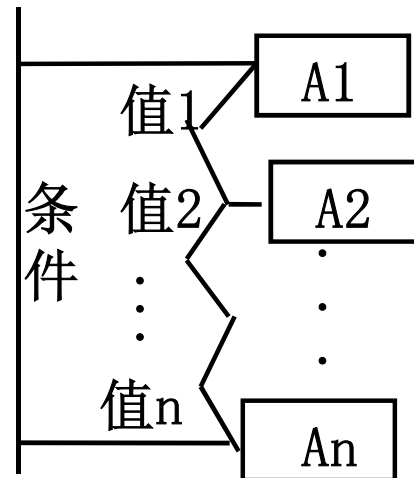
### (先测试循环)



### (后测试循环)



### (4) 多分支选择型 (CASE型)





# 结构化详细设计的工具

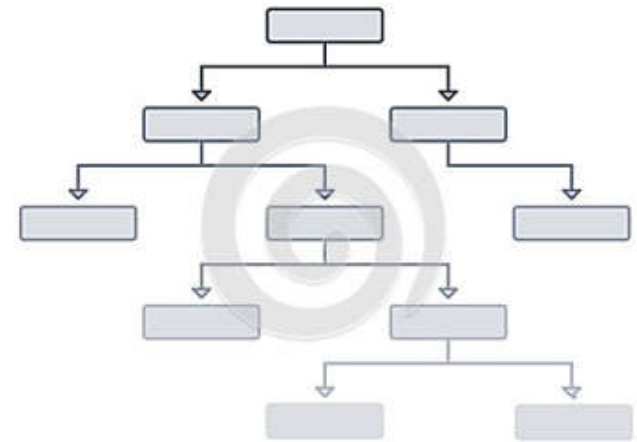
## 会计科目汇总过程 —— PAD图设计

会计信息系统中常将所使用的会计科目及其代码汇集成一张表存到系统里，称为科目汇总表文件。该文件内容也会有变化，所以需要维护，现在请用PAD图来描述维护的作业流程。

其使用流程如下：针对科目汇总文件，打开文件，输入科目代码，通过该代码检索科目。对于检索到的科目，显示该科目记录，并只能在本会计年度内修改该记录。如果是新科目，则追加科目记录。

# 结构化详细设计的工具

**判定树**——用于对复杂条件判断的图形工具。树具有层次结构，树的每个节点表示一个独立的关系表达。随着树节点层次的深入，各节点间的关系组合和理解逐渐复杂。从树根到叶子节点的路径，反映就反映了最终的条件判断的组合。



**判定表**——判定表是对复杂条件判断的表格表示，它清晰地表明所设计的功能是在满足何种条件组合的前提下才被执行。

		列1	列2	.....	列N
条件	条件1				
	条件2				
	.....				
	条件N				
动作	动作1				
	动作2				
	.....				
	动作N				

# 结构化详细设计的工具

某公司规定推销员的薪酬与业绩挂钩，按推销产品收入额提成，上不封顶，下不保底，费用自理。具体为每月推销额10万元以上（含10万元，下同），回款比例达80%且推销的新产品占5成以上者，按推销额的6%提成；新产品不足5成则按5%提成；若回款比例在40%~80%之间，如果新产品占5成以上按5%提成，否则按4%提成；若回款比例低于40%，则按3%提成。推销额不足10万，回款比例在80%以上者则按4%提成；回款比例在40%~80%之间则按3%提成，不足40%则按2%提成。

条件	符号	符号含义
推销额(TXE)	C	超过10万元/月
	D	低于10万元/月
回款(HK)	G	超过80%
	Z	40%~80%之间
	B	不足40%
新产品(XCP)	X	新产品占50%以上
	L	新产品不足50%

# 结构化详细设计的工具

## 详细设计工具 -- 总结

纵观已有的符号体系，如果结合自身的实践经验提出新的符号工具，一般应具有以下的广泛性特征：

- (1) 面向代码
- (2) 控制逻辑
- (3) 模块化设计
- (4) 对结构化设计的支持
- (5) 数据表示
- (6) 易测试
- (7) 易于代码的机器自动生成