4结构化设计方法（5*）

4.1结构化设计方法概述

- 结构化设计的基础是模块
- 结构化设计的基本思想是：基于模块独立性和信息隐藏原则，自顶向下，逐步求精，分解和抽象相结合，并应用结构化程序设计技术而进行的软件设计。
- 1.变换分析法
- 2.事务分析法
- 3.混合分析法

4.2面向数据流的设计方法（5*）

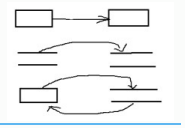
4.2.1层次图和结构图

- 层次图
- 结构图
  - 在层次图的基础上增加了对连线的数据流描述
  - 传入模块接收系统的输入数据，也表示系统内部有先后顺序的模块间的数据传递
  - 传出模块传递系统的输出数据，也表示系统内部有先后顺序的模块间的数据传递
  - 变换模块是系统的功能模块，实现数据的改变
  - 协调模块是系统控制或数据传递模块

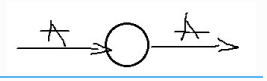
4.2.2变换分析法

第一步：复审并精细化数据流图

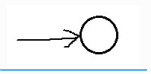
- （1）命名时尽量使用有明确含义的词、短语、术语和领域词汇，减少出现数据流图歧义。
- **（2）上下层图（父子图）在输入、输出以及文件访问数据流之间的平衡。** 错例
  - （1）根据数据守恒原则，外部实体和外部实体，外部实体和数据存储之间不能存在数据流，，存储与存储之间也不应该有数据流，数据流必须跟加工有关，没有加工数据流不可能流来流去的。



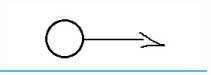
  - （2）对于加工，输入是A，输出还是A，也违反了数据守恒原则，输入与输出一样，加工没有作用。



  - （3）对于加工，只有输入没有输出，违反了数据守恒原则。比如，人不可能只吃饭，不大小便。嘿嘿



  - （4）对于加工，只有输出没有输入，违反了数据守恒原则。比如，人不可能一直大小便，但不吃饭。嘿嘿



- **（3）上下层图（父子图）的层次编号要一致，正确反映数据流图的分解过程。**
- （4）对于每层数据流的分解，可以用逻辑运算符*+⊕增加数据流中各变换部分间的部分
- （5）精化数据流图，使其能使用正确、完整的描述用户需求，因为这将决定软件结构图的逻辑框架正确与否。
- **（6）第一步工作依赖于软件需求规格说明**

第二步：划分自动化边界，确定数据流特征，判断数据流是变换流还是事务流　变换流的特征是数据有明显的输入和输出，处理部分没有过多的控制和判断。

第三步：划分数据输入输出边界，分理出处理部分

第四步：执行一级分解

- 导出软件逻辑结构最上层的两层关系，顶层为主控模块，第二层为根据自动化边界的划分
- 分为以下三个模块
  - 输入模块
  - 输出模块
  - 控制模块
- 一级分解的总原则是抽象，在能明确表示软件总体框架和各部分逻辑子系统的前提下，模块数应尽量少。

第五步：执行二级分解

二级分解的任务是把一层分级得到的各子系统模块按照各层的数据流图逐层细分，得到系统结构图的原型

第六步：采用启发式规则，精化所得到的初步软件结构，以模块独立性为原则，合并、分解、抽取各个模块，**得到**一个高内聚、低耦合、易实现、易测试、易维护的**软件结构图**

4.2.3事务分析法

- 当数据例图具有事务特征，即能找到事务中心和对应的多条活动路径的，则采用事务分析法。
- 事务流的特征是在数据的输入、处理和输出的过程中，处理部分有明显的控制或判断中心，后续的数据流有较多的活动路径。

4.2.4混合分析法

4.4结构化详细设计的工具（5*）

- 程序流程图
  - 优点：易懂，使用广泛
  - 缺点：
    - 对程序流程图中的控制流无法约束其转向，造成设计的随意性，并可能造成非结构化设计的出现。
    - 难以表达数据结构
- 盒图
  - 没有控制流，控制域明晰
  - 盒图中方框的互相嵌套，准确的反映了过程设计时模块间的层次关系
- 问题分析图（PAD）
- 判定树
- 判定表
- 重点在于会画图