系统设计

51CTO学院：邹月平

系统设计

软件设计包括体系结构设计、接口设计、数据设计和过程设计。

体系结构设计	定义软件系统各主要部件之间的关系。
数据设计	将模型转换成数据结构的定义。好的数据设计将改善程序结构和模块划分，降低过程复杂性。
接口设计（人机界面设计）	软件内部，软件和操作系统间以及软件和人之间如何通信。
过程设计	系统结构部件转换成软件的过程描述。

典型真题

● 软件设计包括了四个既独立又相互联系的活动：高质量的（ ）将改善程序结构和模块划分，降低过程复杂性；（ ）的主要目标是开发一个模块化的程序结构，并表示出模块间的控制关系；（ ）描述了软件与用户之间的交互关系。

A. 程序设计　　B. 数据设计　　C. 算法设计　　D. 过程设计

A. 软件结构设计　　B. 数据结构设计　　C. 数据流设计　　D. 分布式设计

A. 数据架构设计　　B. 模块化设计　　C. 性能设计　　D. 人机界面设计

典型真题

试题分析

软件设计包括体系结构设计、接口设计、数据设计和过程设计。

结构设计：定义软件系统各主要部件之间的关系。

数据设计：将模型转换成数据结构的定义。好的数据设计将改善程序结构和模块划分，降低过程复杂性。

接口设计（人机界面设计）：软件内部，软件和操作系统间以及软件和人之间如何通信。

过程设计：系统结构部件转换成软件的过程描述。

参考答案：B 、A 、D

结构化设计

1 概要设计

确定每个模块的功能和调用关系，形成软件的模块结构图，即SC系统结构图。

确定结构

2 详细设计

为每个具体模块选择适当的技术手段和处理方法的过程。

实现细节

结构化设计

初始系统结构图

结构化设计

系统结构图（Structure Chart，SC）又称为模块结构图，它是软件概要设计阶段的工具，反映系统的功能实现和模块之间的联系与通信，包括各模块之间的层次结构，即反映了系统的总体结构。

SC包括模块、模块之间的调用关系、模块之间的通信和辅助控制符号等四个部分。

（1）模块。在SC中，模块用矩形框表示，框中标注模块的名字，对于已定义或者已开发的模块，可以用双纵边矩形框表示。

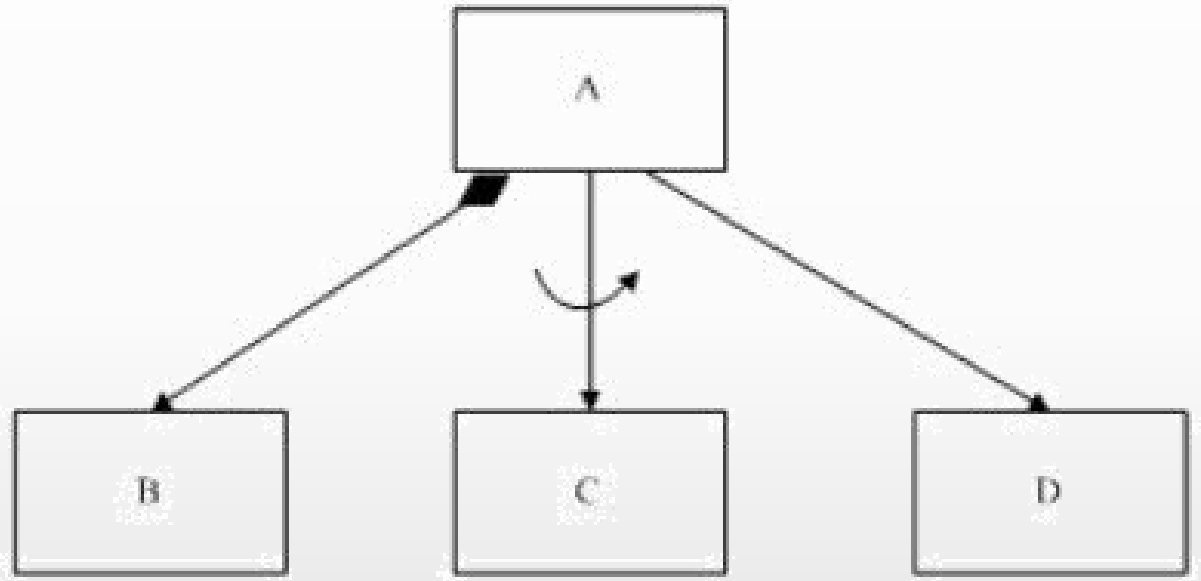
结构化设计

（2）模块之间的调用关系。绘制方法是两个模块一上一下布局，以箭头相连，上面的模块是调用模块，箭头指向的模块是被调用的模块。

（3）模块之间的通信。模块间的通信以表示调用关系的长箭头旁边的短箭头表示，短箭头的方向和名字分别表示调用模块和被调用模块之间信息的传递方向和内容。

（4）辅助控制符号。当模块A有条件地调用模块B时，在箭头的起点标以菱形，模块A反复调用模块C时，在调用关系的连线上增加一个环状的箭头。在SC中，条件调用时所依赖的判断条件和循环调用时所依赖的控制条件通常都无须注明。

结构化设计

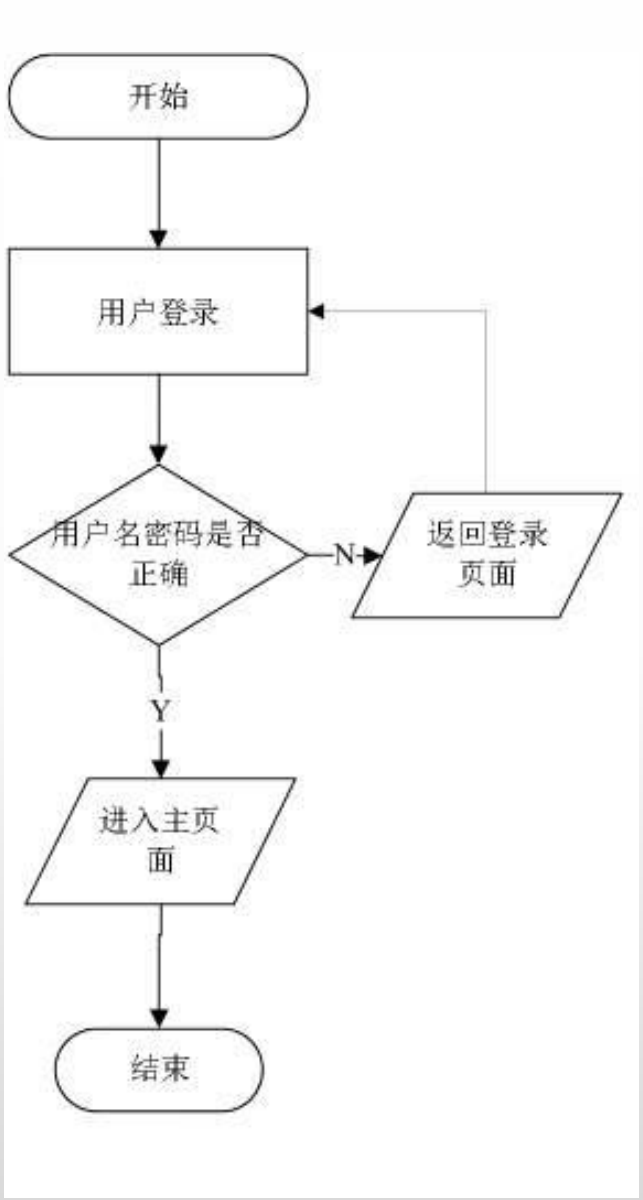


流程设计

包括程序流程图、IPO图、盒图、问题分析图、判定树，表格工具包括判定表，语言工具包括过程设计语言等。

流程设计

1、程序流程图

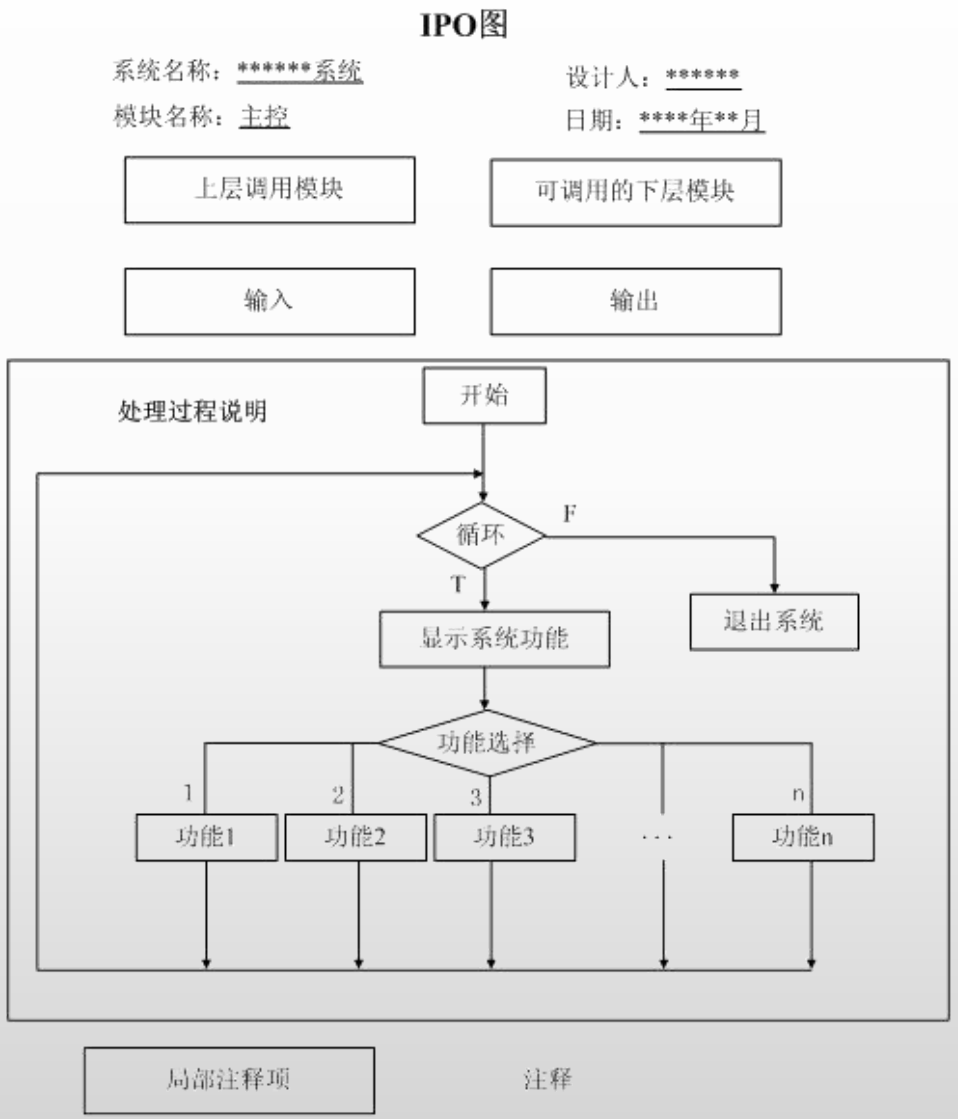


流程设计

2. IPO图

系统分析阶段产生的数据流图经转换和优化后形成的系统模块结构图的过程中将产生大量的模块，分析与设计人员应为每个模块写一份说明，即可用IPO图来对每个模块进行表述，IPO图用来描述每个模块的输入、输出和数据加工。

流程设计

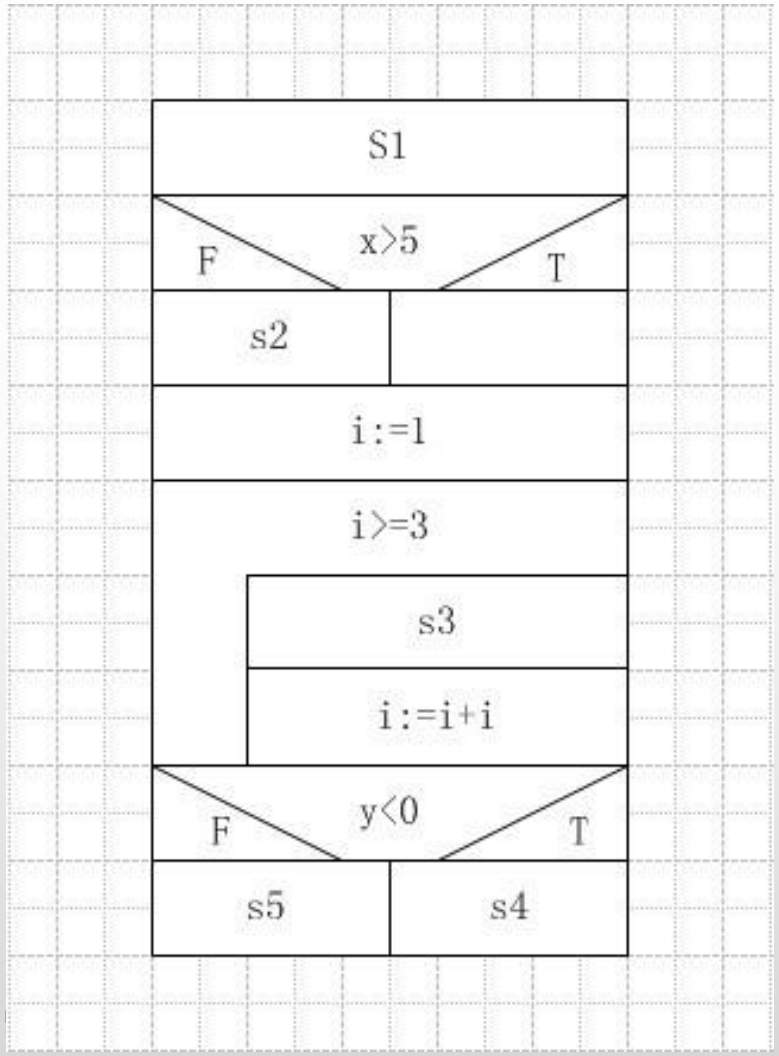


流程设计

3. N-S图

在N-S图中也包括五种控制结构，分别是顺序型、选择型、WHILE循环型（当型循环）、UNTIL循环型（直到型循环）和多分支选择型，任何一个N-S图都是这五种基本控制结构相互组合与嵌套的结果。 N-S图避免了流程图在描述程序逻辑时的随意性与灵活性。

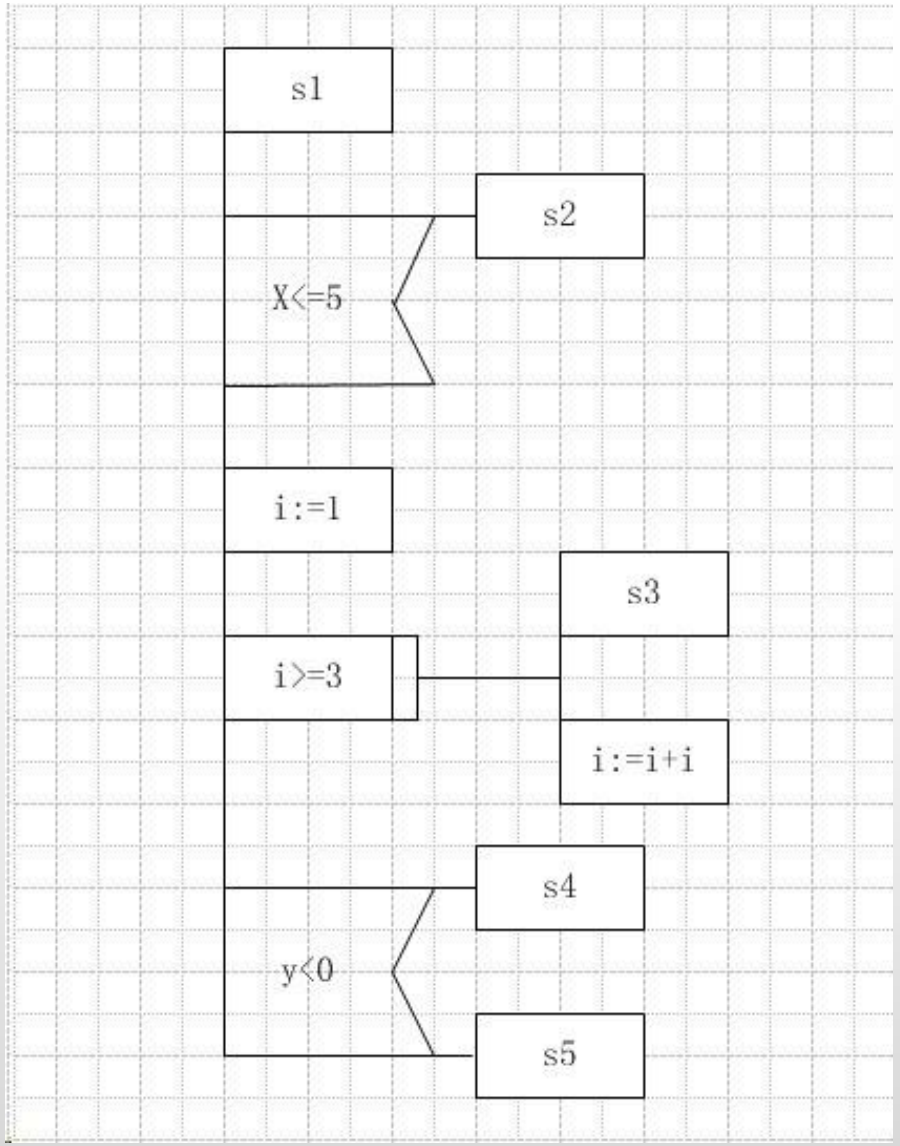
流程设计



流程设计

4. 问题分析图

问题分析图（Problem Analysis Diagram，PAD）包含五种基本控制结构，并允许递归使用。



流程设计

5. 过程设计语言

过程设计语言（Process Design Language，PDL）也称为结构化语言或伪代码（pseudo code），它是一种混合语言，采用自然语言的词汇和结构化程序设计语言的语法，用于描述处理过程怎么做，类似于编程语言。

流程设计

```
寻找物流商{
    WHILE（有新订单）
    DO{
        IF 订单.类型==′普通二手车′ AND 订单.路线复合固定路线或包车路线
                THEN 自动分配给合约物流商；
                ELSE 分配订单到竞拍体系
            ENDIF
        给承运的物流商发送信息；
        更新订单的物流信息；
        给车辆交易系统发送物流信息；
      }ENDDO
}
```

流程设计

6. 判定表

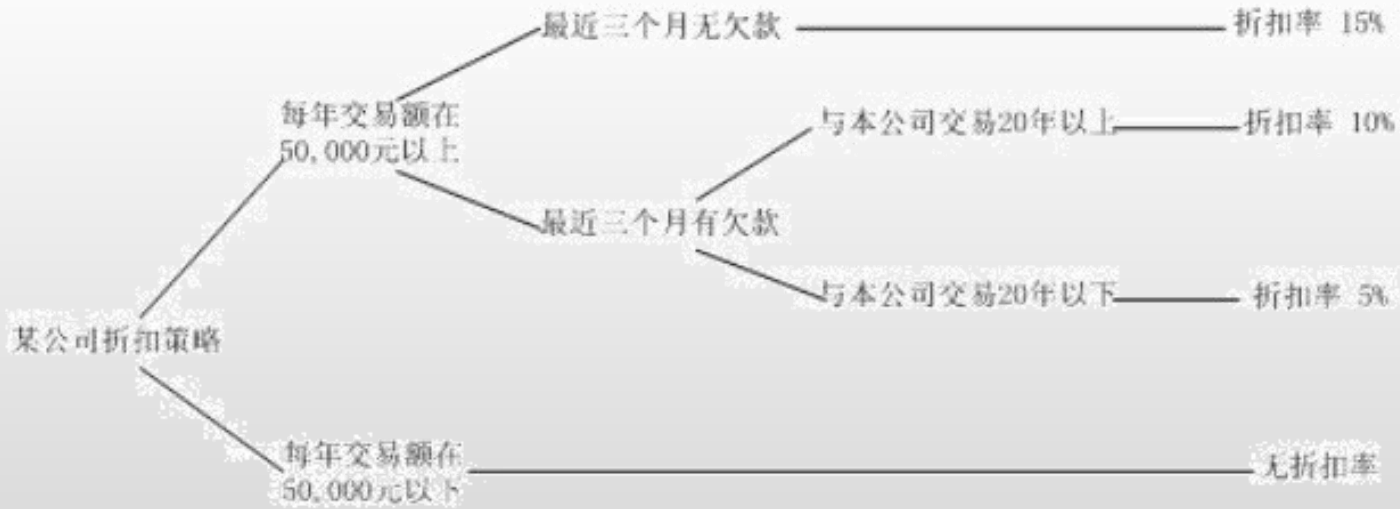
对于具有多个互相联系的条件和可能产生多种结果的问题，用结构化语言描述则显得不够直观和紧凑，这时可以用以清楚、简明为特征的判定表（decision table）来描述。

不同条件组合 / 条件和行动	1	2	3	4	5	6	7	8
C1：每年交易额在 50,000 以上	T	T	T	T	F	F	F	F
C2：最近三个月无欠款	T	T	F	F	T	T	F	F
C3：与本公司交易 20 年及以上	T	F	T	F	T	F	T	F
A1：折扣率 15%	Y	Y						
A2：折扣率 10%			Y					
A3：折扣率 5%				Y				
A4：无折扣率					Y	Y	Y	Y

流程设计

7. 判定树

判定树（decision tree）也是用来表示逻辑判断问题的一种常用的图形工具，它用树来表达不同条件下的不同处理流程，比语言、表格的方式更为直观。判定树的左侧（称为树根）为加工名，中间是各种条件，所有的行动都列于最右侧。



面向对象设计

单一职责原则	设计目的单一的类。
开放-封闭原则	对扩展开放，对修改封闭。
里式替换原则	子类可以替换父类
依赖倒置原则	要依赖于抽象，不是具体实践。对接口进行编程，不要对实现编程。
接口隔离原则	使用多个专门的接口比使用单一的总接口好。
组合重用原则	尽量使用组合不是继承达到重用的目的、
迪米特原则（最少知识）	一个对象应当对其他对象有尽可能少的了解。

典型真题

● 在面向对象设计的原则中，（ ）原则是指抽象不应该依赖于细节，细节应该依赖于抽象，即应针对接口编程，而不是针对实现编程。

A．开闭　　B．里氏替换　　C．最少知识　　D．依赖倒置

典型真题

试题分析

依赖倒置原则是指抽象不应该依赖于细节，细节应当依赖于抽象。换言之，要针对接口编程，而不是针对实现编程。在程序代码中传递参数时或在组合(或聚合)关系中，尽量引用层次高的抽象层类，即使用接口和抽象类进行变量类型声明、参数类型声明和方法返回类型声明，以及数据类型的转换等，而不要用具体类来做这些事情。为了确保该原则的应用，一个具体类应当只实现接口和抽象类中声明过的方法，而不要给出多余的方法，否则，将无法调用到在子类中增加的新方法。

典型真题

实现开闭原则的关键是抽象化，并且从抽象化导出具体化实现，如果说开闭原则是OOD的目标的话，那么依赖倒置原则就是OOD的主要机制。有了抽象层，可以使得系统具有很好的灵活性，在程序中尽量使用抽象层进行编程，而将具体类写在配置文件中，这样，如果系统行为发生变化，则只需要扩展抽象层，并修改配置文件，而无须修改原有系统的源代码，在不修改的情况下来扩展系统功能，满足开闭原则的要求。依赖倒置原则是COM、CORBA、EJB、Spring等技术和框架背后的基本原则之一。

参考答案：D

技术成就梦想