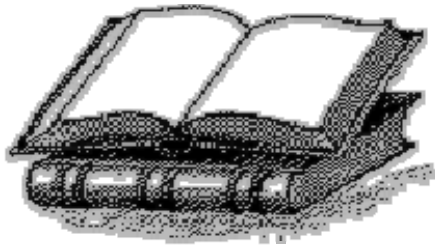
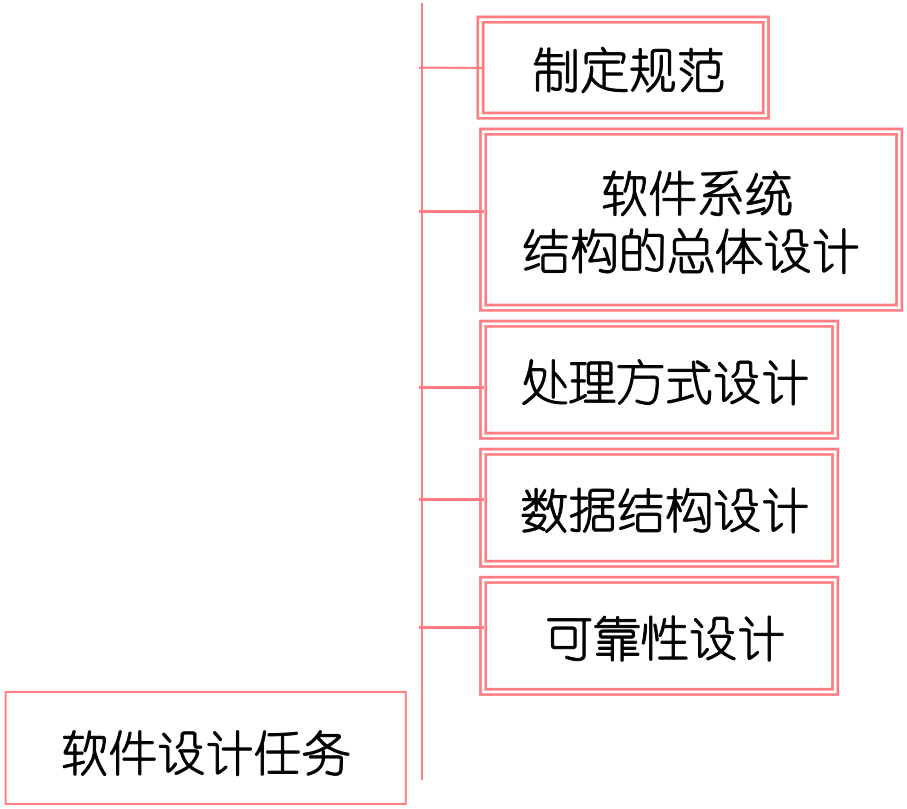
《软件工程》

Software Engineering

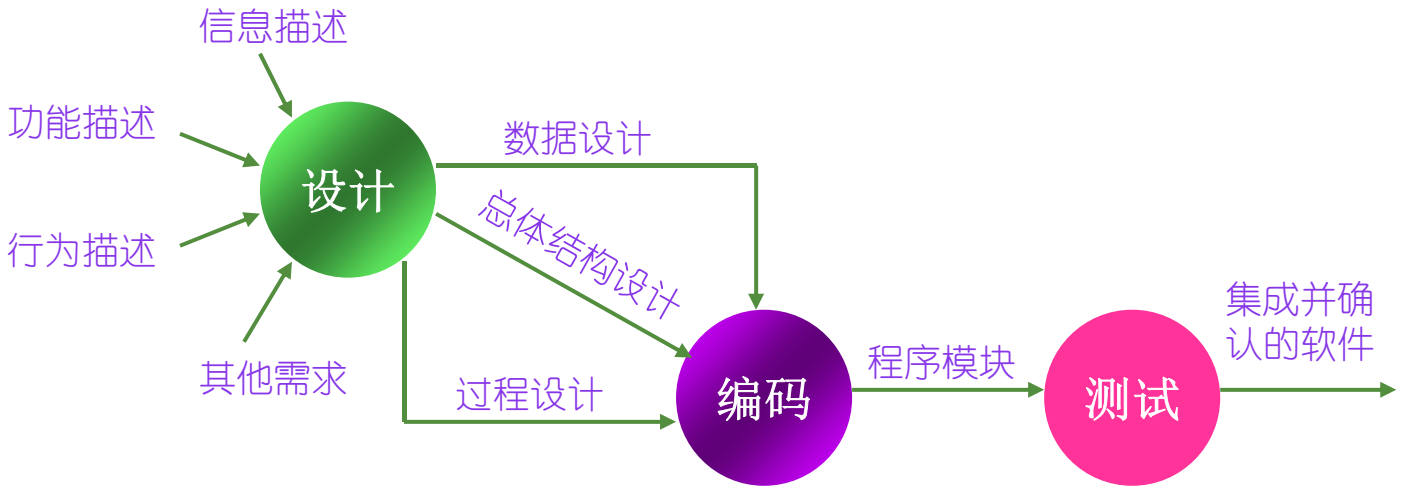


2019/9/25

第四章 结构化设计



4.1 软件设计过程

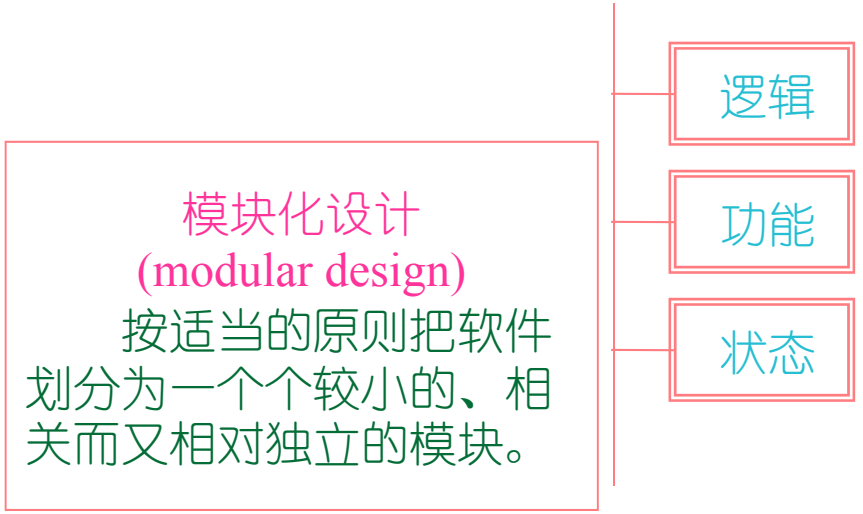


------ 软件开发阶段的信息流

4.2 软件设计的概念与原理

一、模块化设计与信息隐藏

模块(module)-----“模块“又称”构件”一般指用一个名字调用的一段程序



1、分解（decomposition）

奇妙的数字 7±2，人类信息处理能力的限度

G.A. Miller

Magical Number Seven,Plus or Minus Two,Some Limits on Our Capacity for Processing Information

The Psychological Review,1956

设：C(x) 为复杂程度函数

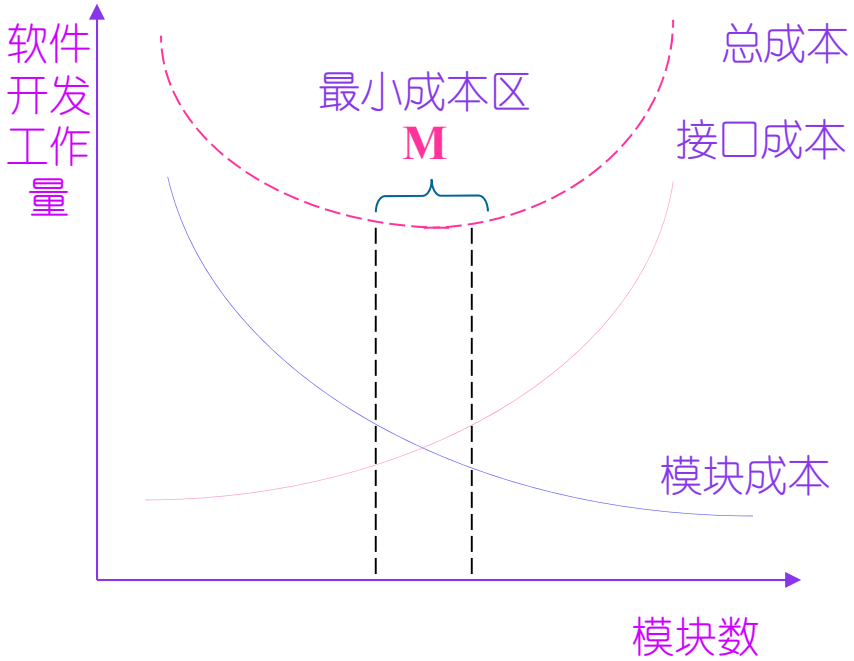
E(x) 为决定解决问题x所需的工作量(时间)函数

C(P1)>C(P2)　　E(P1)>E(P2)

C(P1+P2)>C(P1)+C(P2)　　E(P1+P2)>E(P1)+E(P2)

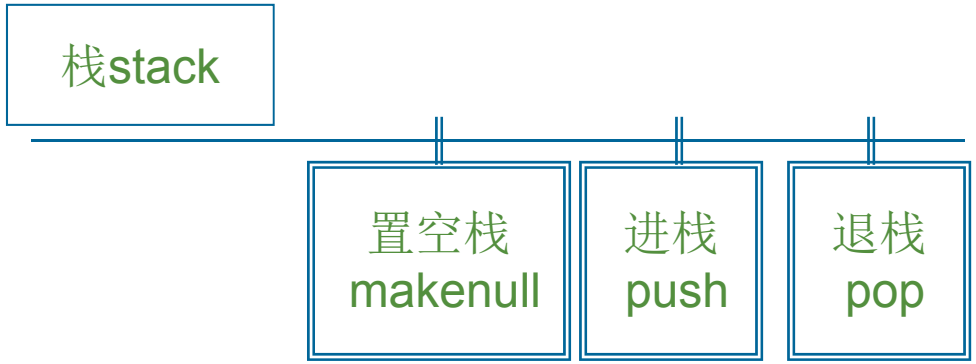
软件工程基本定理

C(P1+P2)>C(P1)+C(P2)
E(P1+P2)>E(P1)+E(P2)



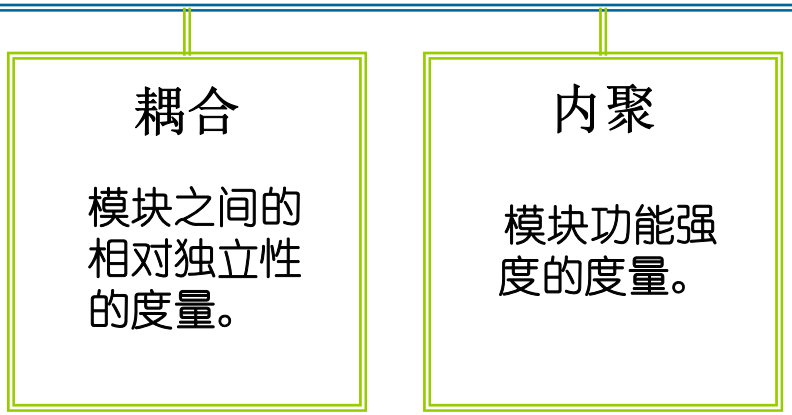
2、信息隐藏(information hiding)

每个模块的实现细节对于其他模块来说是隐藏的。也就是说，模块中所包含的信息是不允许其他不需要这些信息的模块使用的。



3、模块的独立性(module independence)

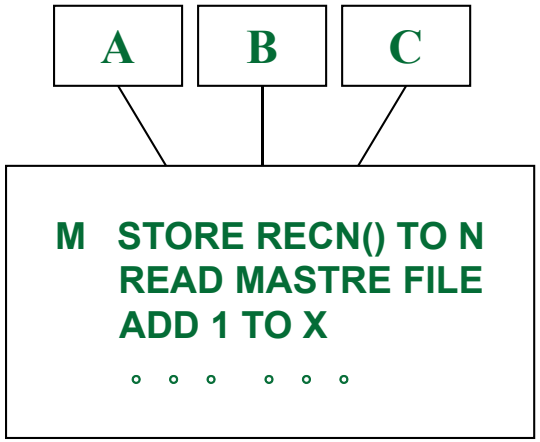
模块的独立性是指软件系统中每个模块只涉及软件要求的具体的子功能，而和软件系统中其他模块的接口是简单的。



(1) 内聚(Cohesion)

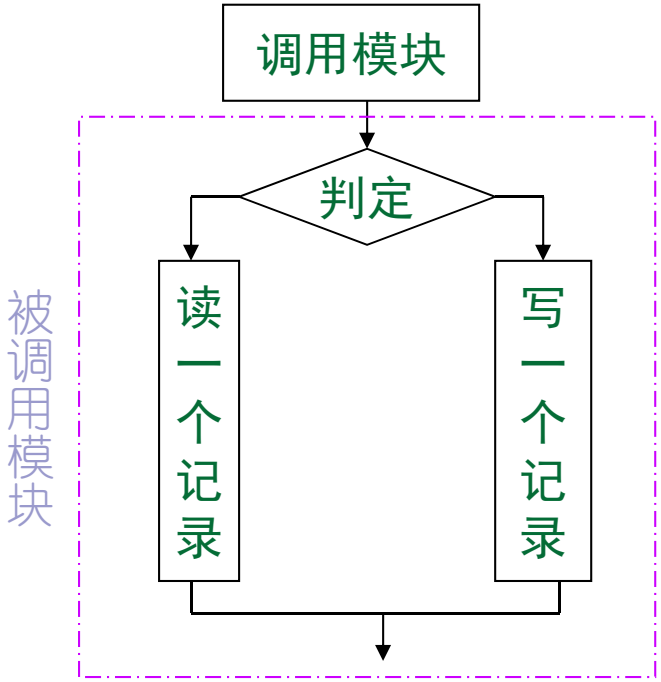
偶然性内聚

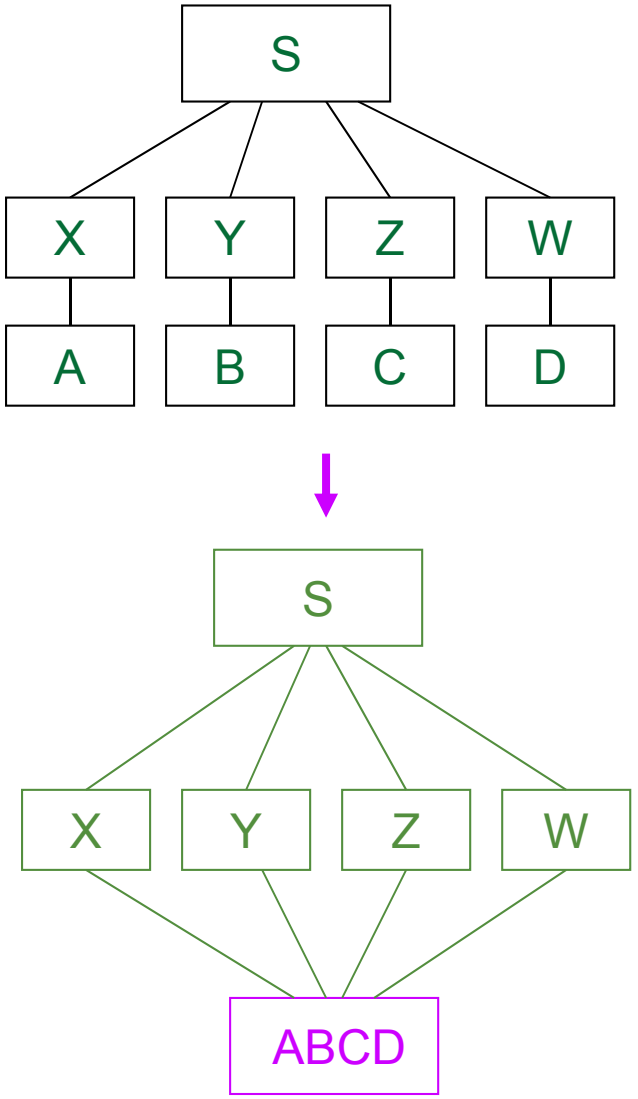
当模块内各部之间没有联系，或者即使有联系，这种联系也很松散。则称这种模块为巧合内聚模块。



逻辑性内聚

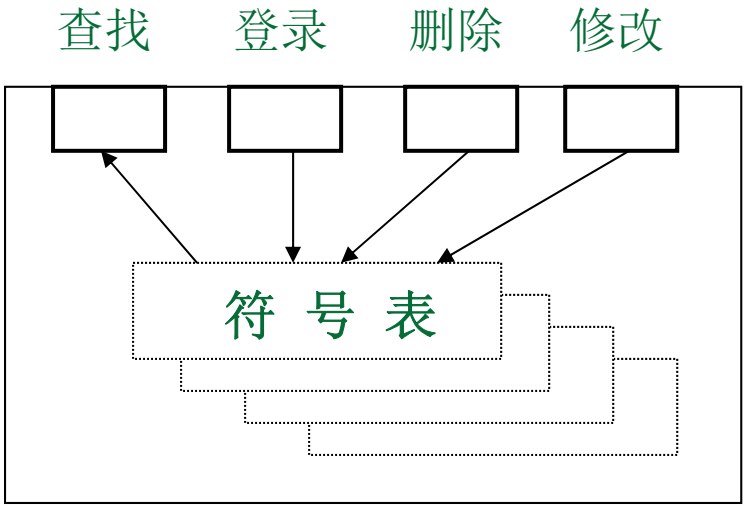
这种模块是把几种功能组合在一起，每次调用时，则由传递给模块的判定参数来确定该模块应执行哪一种功能。




S
X
Y
Z
W
A
B
C
D
S
X
Y
Z
W
ABCD

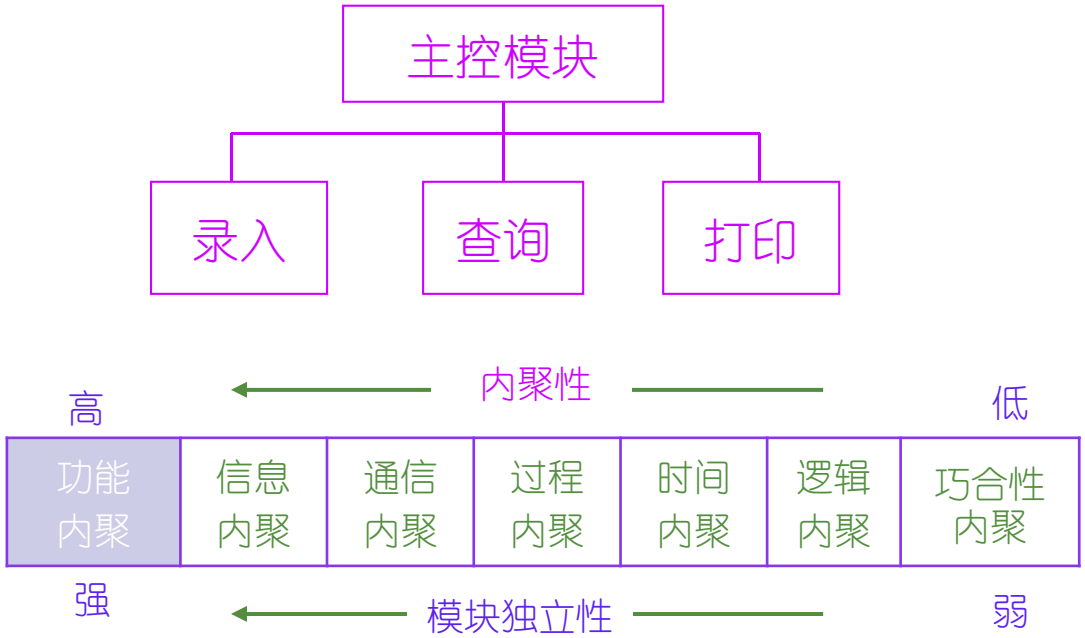
信息性内聚

这种模块能完成多个功能，各个功能都在同一数据结构上操作，每一项功能有一个唯一的入口点。



功能性内聚

如果一个模块内所有成分都完成一个功能则称这样的模块为功能模块。

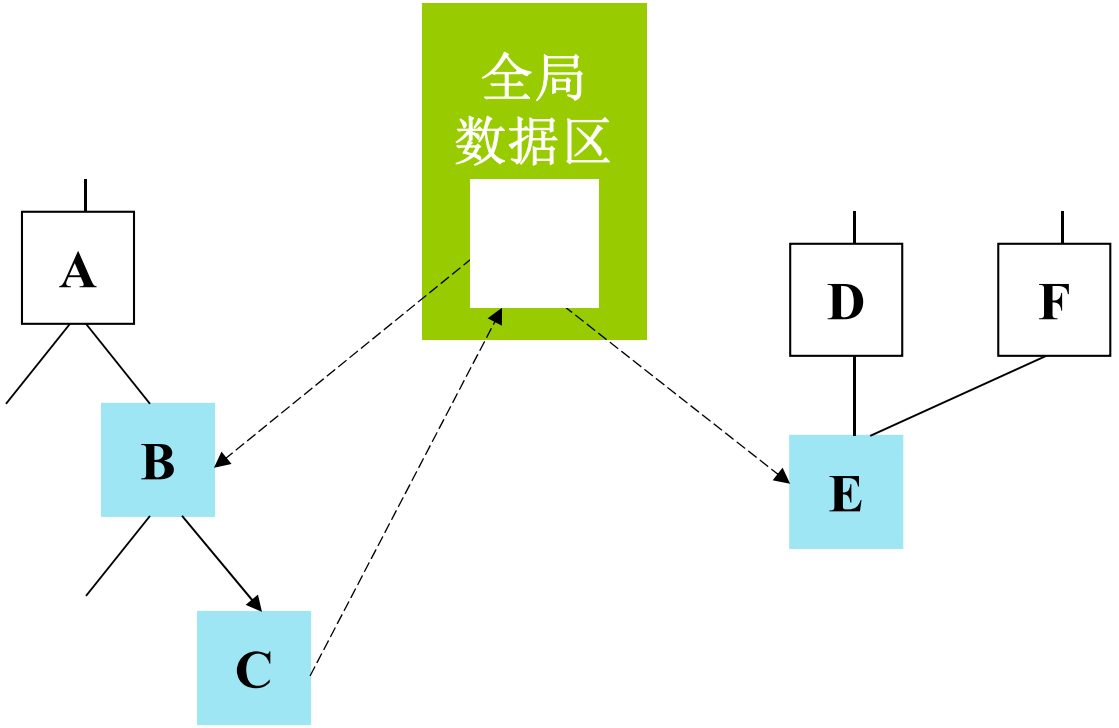


(2) 耦合

耦合性是程序结构中各个模块之间相互关联的度量它取决于各个模块之间接口的复杂程度、调用模块的方式以及那些信息通过接口。

公共耦合

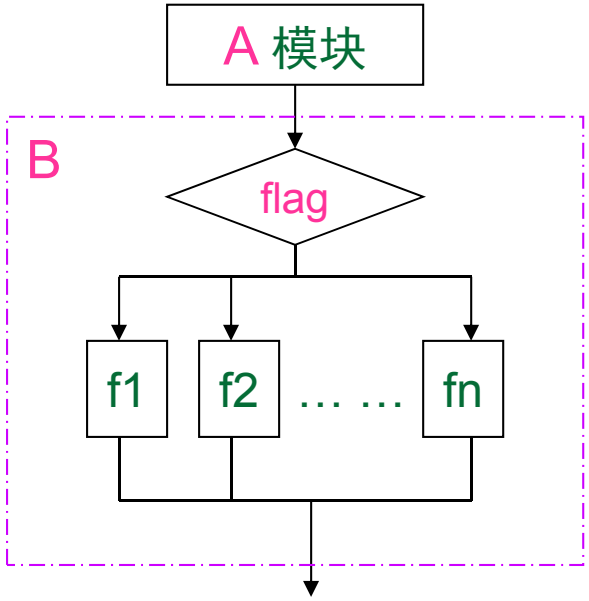
允许一组模块访问同一全局性的数据结构。



B、C、E 为公共耦合

控制耦合

如果一个模块通过传送开关、标志、名字等控制信息，明显地控制选择另一模块的功能就是控制耦合。



以上给出了7种耦合类型，这只是从耦合的机制上所做的分类，按耦合的强弱程度的排列只是相对的关系。但它给设计人员在设计程序结构时提供了一决策准则。实际上，开始时两个模块之间的耦合不只是一种类型，而是多种类型的混合。这就要求设计人员按照实际情况进行分析、比较和分析，逐步加以改进，以提高模块的独立性。

设计方法核心：抽象

- 今有雉兔同笼，上有三十五头，下有九十四足，问雉兔各几何？
- 程序员果冻觉得写程序赚钱不多，他想捞外快。于是他参加了某村的搬砖大队， 大队规定搬砖到目的地，没有破损则给运费每块砖四分钱，如果有任何破损或丢失则倒扣一毛五分钱。最后他搬了 一千块砖，共得三十五块两毛五分钱。问果冻搬的砖头没有破损的有多少块？

抽象为同一问题

鸡兔同笼：

$x+y=35;$　　$4*x+2*y=94$

果冻搬砖：

$x+y=1000;$　　$4*x-15*y=3525$

4.3 结构化设计方法(SD-- Structured Design)

结构化设计方法是基于模块化、自顶向下细化、结构化程序设计等程序设计技术基础发展起来的。

它所提供的方法和原则，主要是用来指导软件的概要设计。它还提供了一种 “结构图” 的描述工具，是专门用来描述软件的总体结构的。

结构化设计属于面向数据流 的设计方法。

在软件的需求分析阶段，数据流是软件开发人员考虑问题的出发点和基础。数据流从系统的输入端向输出端，则要经历一系列的变换或处理。用来表现这个过程的数据流（**DFD**），实际上就是软件系统的逻辑模型。

面向数据流的设计要解决的任务，就是在上述需求分析的基础上，将DFD图 映射（Mapping）--- **软件系统的结构**。

换句话说，这类设计方法，允许把用 DFD图表示的系统逻辑模型，很方便地转换成对于软件结构的初始设计描述。

结构化设计方法中，软件的结构一律用 **SC** 图来描述。

结构化设计的目的

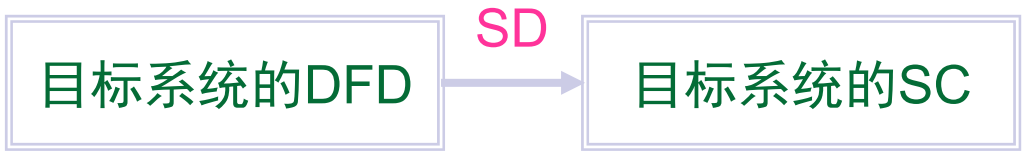
- 使程序的结构尽可能反映要解决的问题的结构

结构化设计的任务

- 完成目标系统的一系统结构图(SC)

SC 图 --- Structured Chart

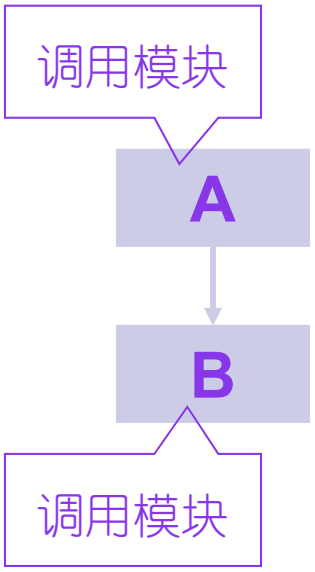
该图常用来表示系统的软件结构。利用它可以清楚地表达软件结构中模块间的层次调用关系和模块之间的联系。



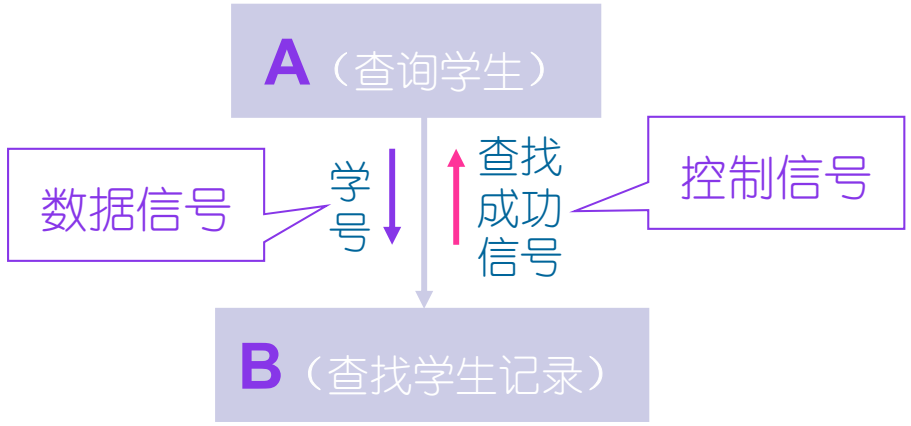
SC 图中的主要内容

1、模块--在SC图中用矩形框表示，并用名字来标记它

2、模块的调用关系和接口

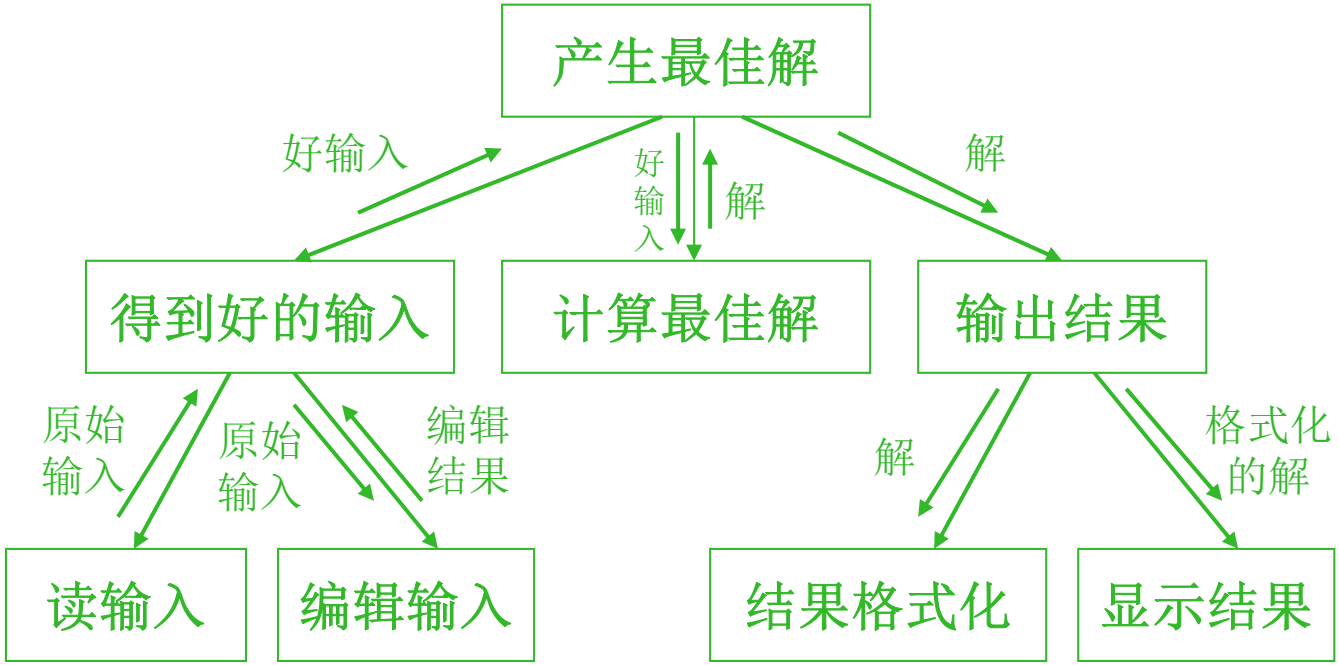


-- 模块调用关系



-- 模块间接口的表示

SC 图的一般格式


产生最佳解
好输入
好输入
解
解
得到好的输入
计算最佳解
输出结果
原始输入
原始输入
编辑结果
解
格式化的解
读输入
编辑输入
结果格式化
显示结果

一、典型的系统结构形式

1、在系统结构图中的模块

原子模块：在系统结构图中通常是指不能再分割的底层模块

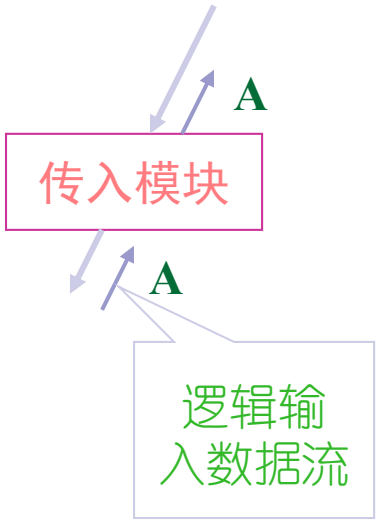
完全因子分解系统

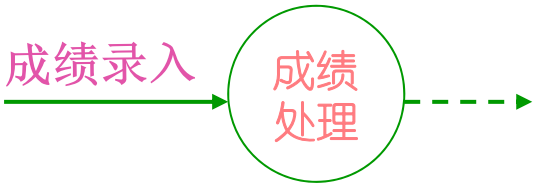
如果一个软件系统，它的全部实际加工(即数据计算或处理)都是由底层的原子模块来完成，而其它所有非原子模块仅仅执行控制或协调功能。

在系统结构图中有四种类型的模块：

- 传入模块

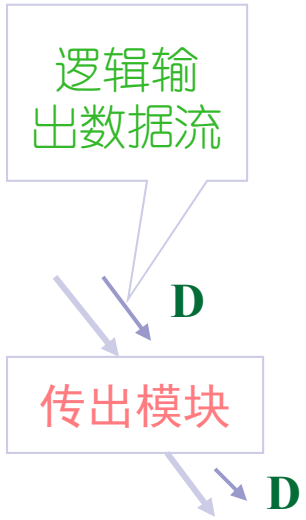
从下属模块取得数据，进行某些处理，再将其结果传给上级模块。在此，将它传送的数据流称为逻辑输入数据流。

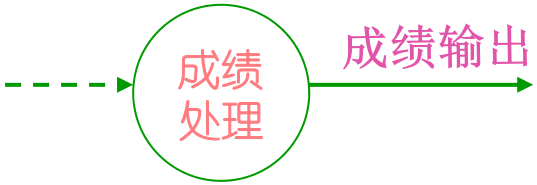




- 传出模块

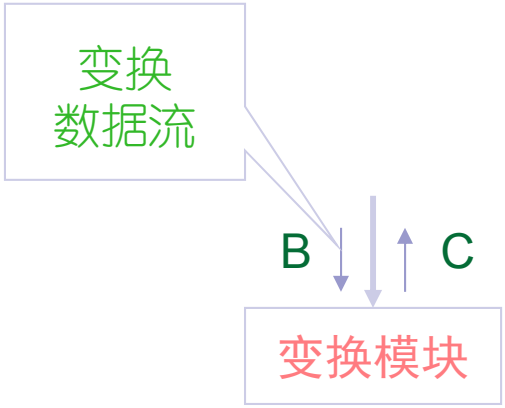
从上级模块获得数据，进行某些处理，再将其结果传给下属模块。在此，将它传送的数据流称为逻辑输出数据流。





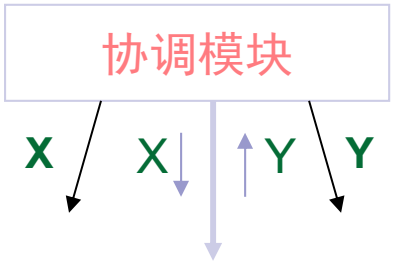
- 变换模块

　　也叫加工模块。它是从上级模块获得数据，进行特定的处理，将其转换为其他形式，再传回上级模块它所加工的数据流叫做变换数据流。

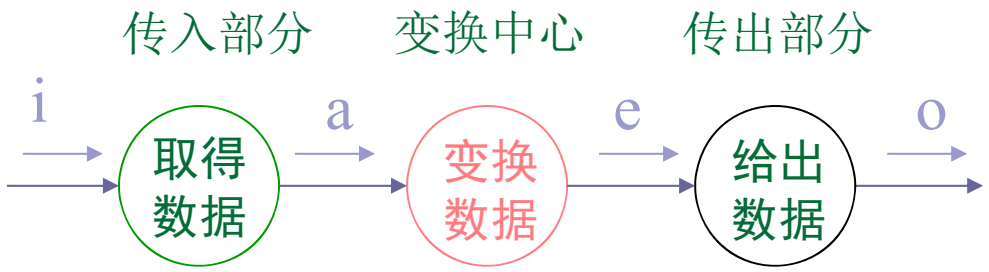


- 协调模块

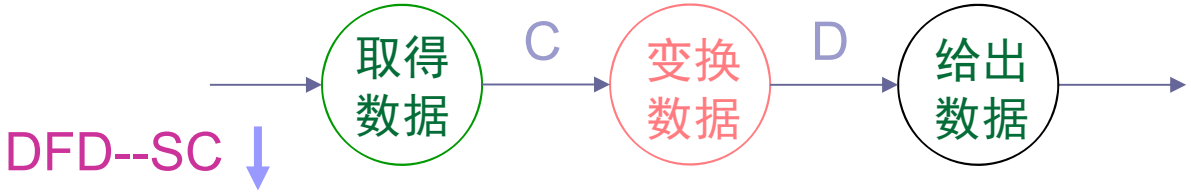
对所有下属模块进行协调和管理的模块。在一个好的系统结构图中，协调模块应在较高层出现。

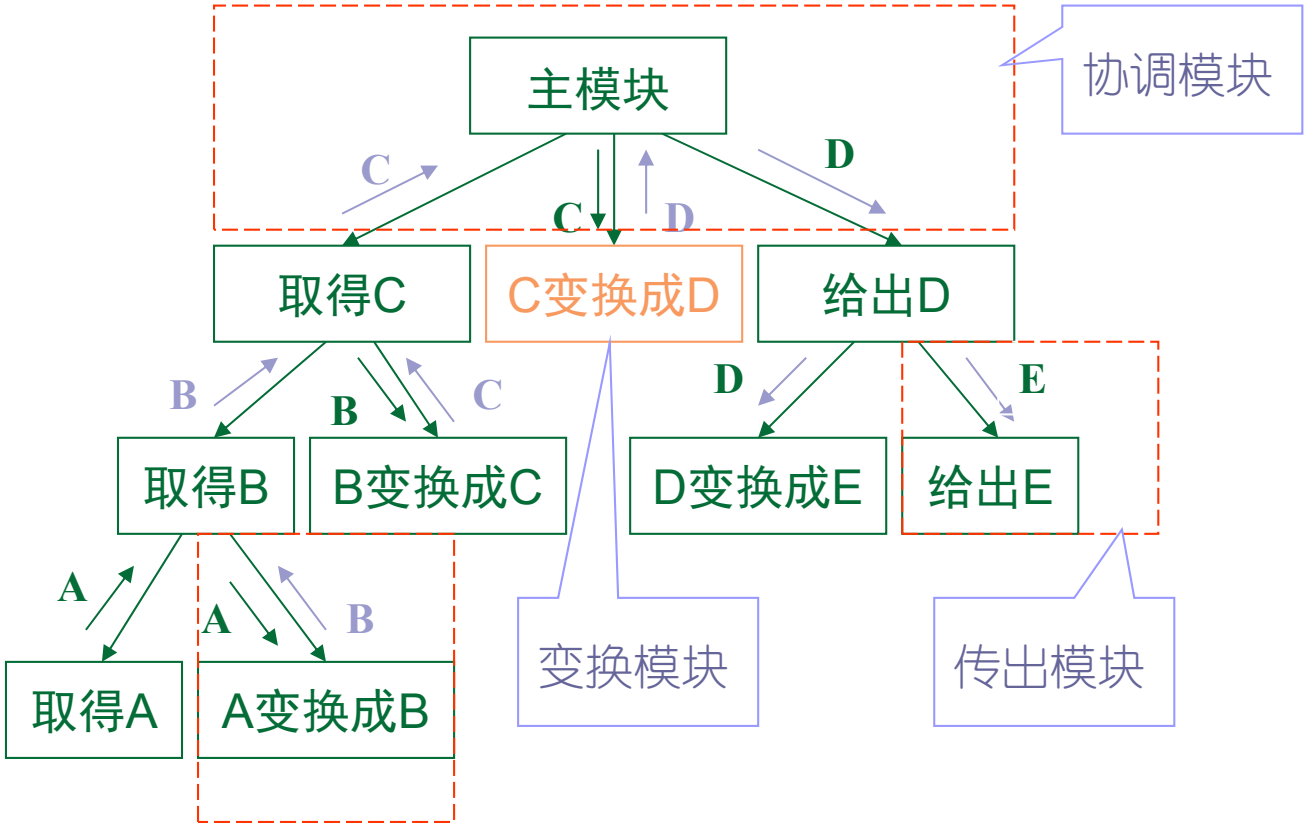


2、典型的系统结构形式之一(变换型系统结构图)



-------- 具有变换型数据流图


取得
数据
C
变换
数据
D
给出
数据
DFD--SC


主模块
协调模块
C
D
C
D
取得C
C变换成D
给出D
B
B
C
D
E
取得B
B变换成C
D变换成E
给出E
A
A
B
取得A
A变换成B
变换模块
传出模块

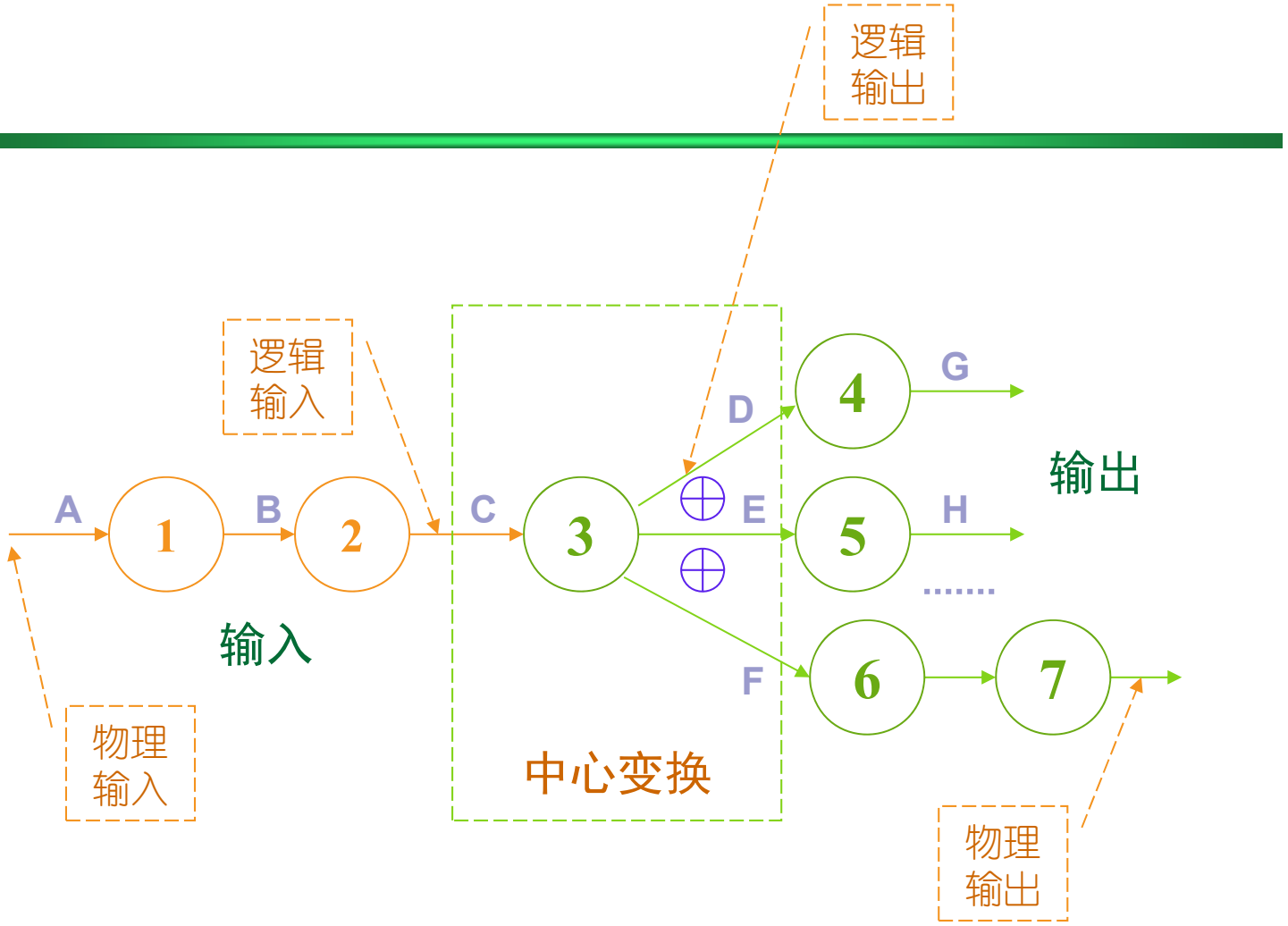
-------- 具有变换型系统结构图

3、典型的系统结构形式之二(事务型系统结构图)

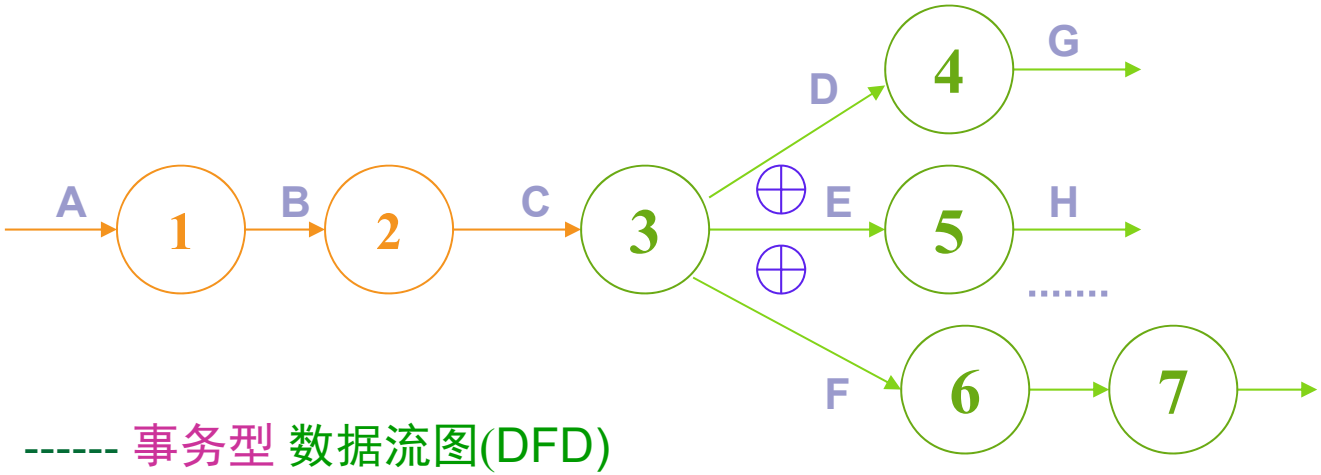
“事务”

引起、触发或启动某一动作或一串动作的任何数据、控制信号、事件或状态的变化。

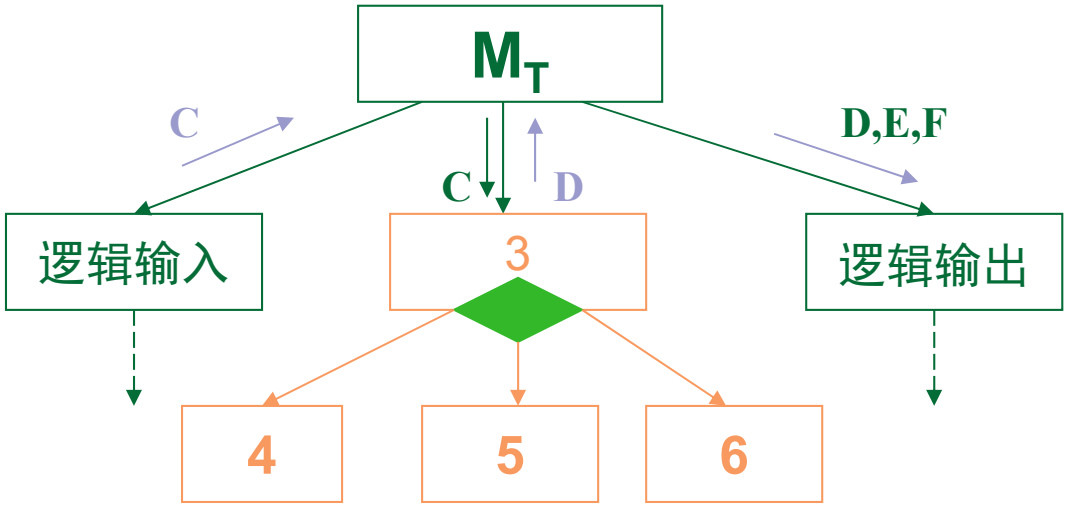
由它接受一项事务，根据事务处理的特点和性质选择分配一个适当的处理单元，然后给出结果。



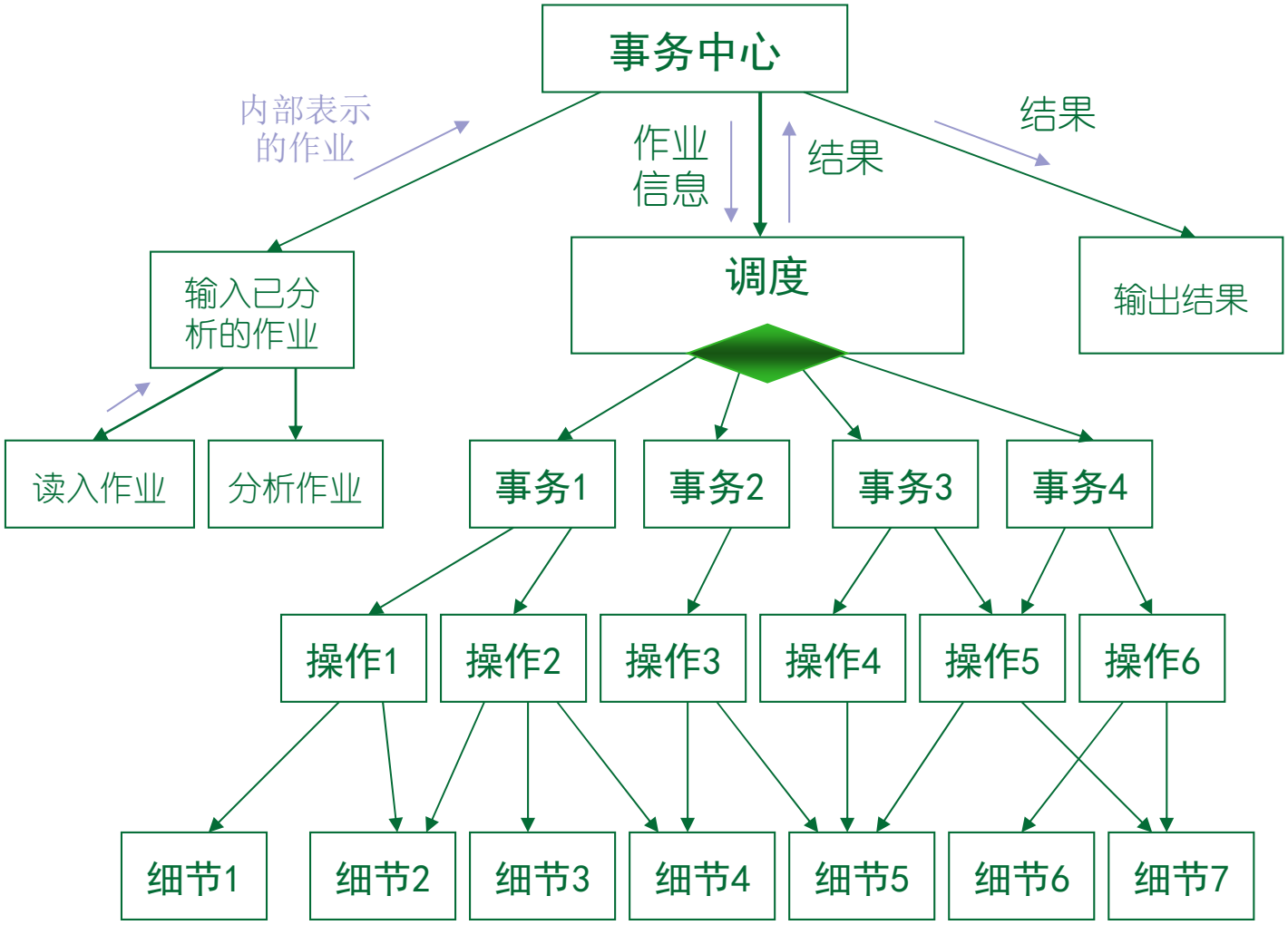
------ 事务型 数据流图



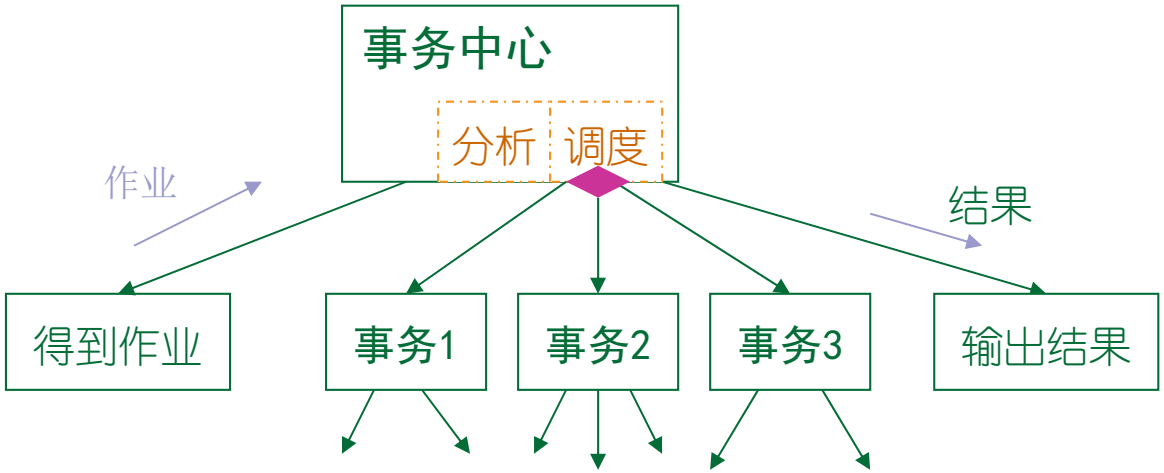
------ 事务型 数据流图(DFD)



------ 事务型 系统结构图(SC)

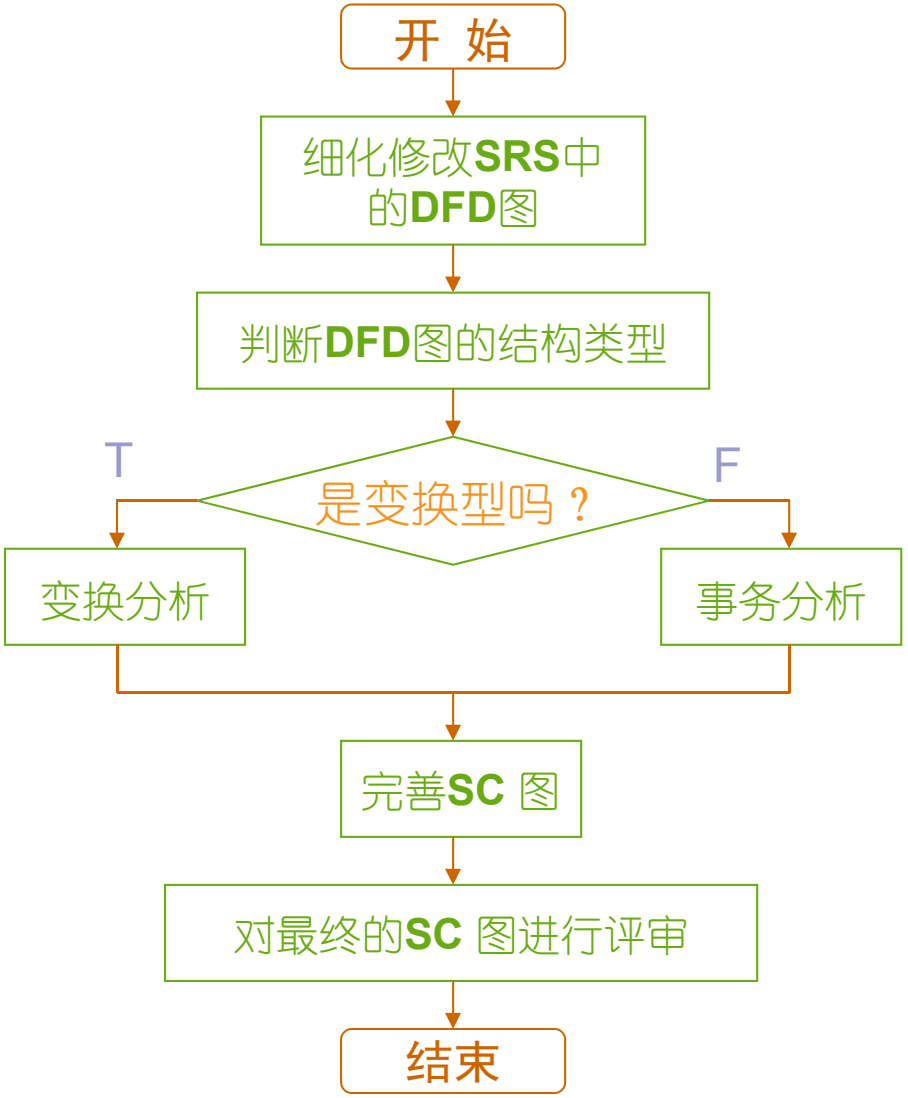

事务中心
内部表示
的作业
作业
信息
结果
结果
输入已分
析的作业
调度
输出结果
读入作业
分析作业
事务1
事务2
事务3
事务4
操作1
操作2
操作3
操作4
操作5
操作6
细节1
细节2
细节3
细节4
细节5
细节6
细节7

------ 事务型系统结构(层次)图



------简化的事务型系统结构图

二、从 DFD 图导出 SC图的步骤

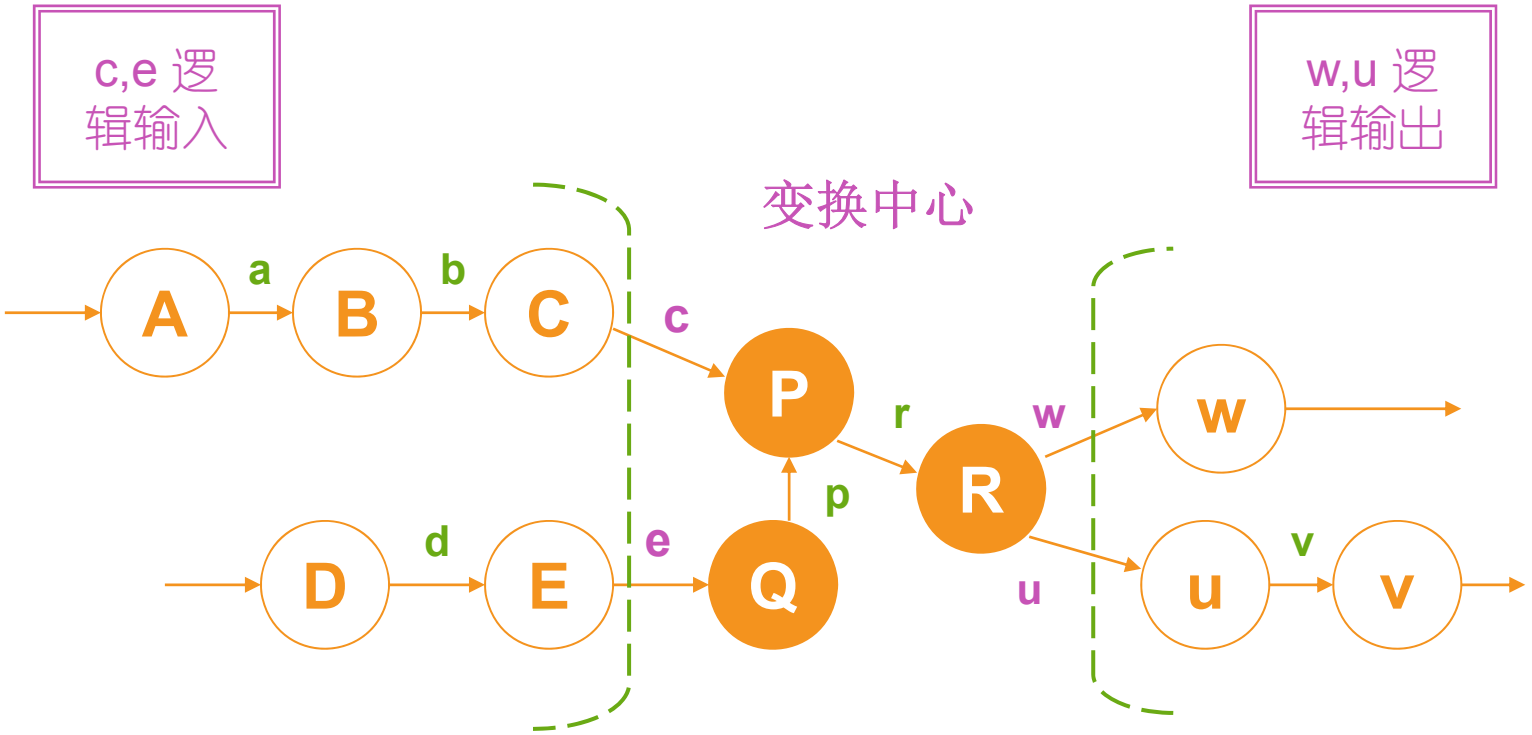


1、变换分析 ----是将具有变换型的DFD图导出SC图

变换分析

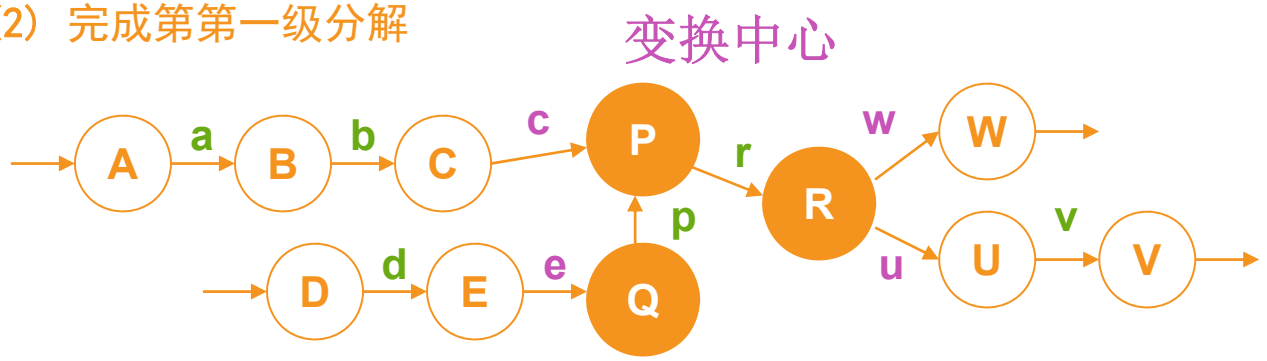
- 从物理输入、物理输出及变换中心进行由顶向下的分解得出各个分支的所有组成模块
- 在数据流图上区分系统的逻辑输入、逻辑输出和变换中心部分，并标出它们的分界。
- 进行一级分解，设计系统模块结构的顶层和第一层。
- 进行二级分解，设计中、下层模块。

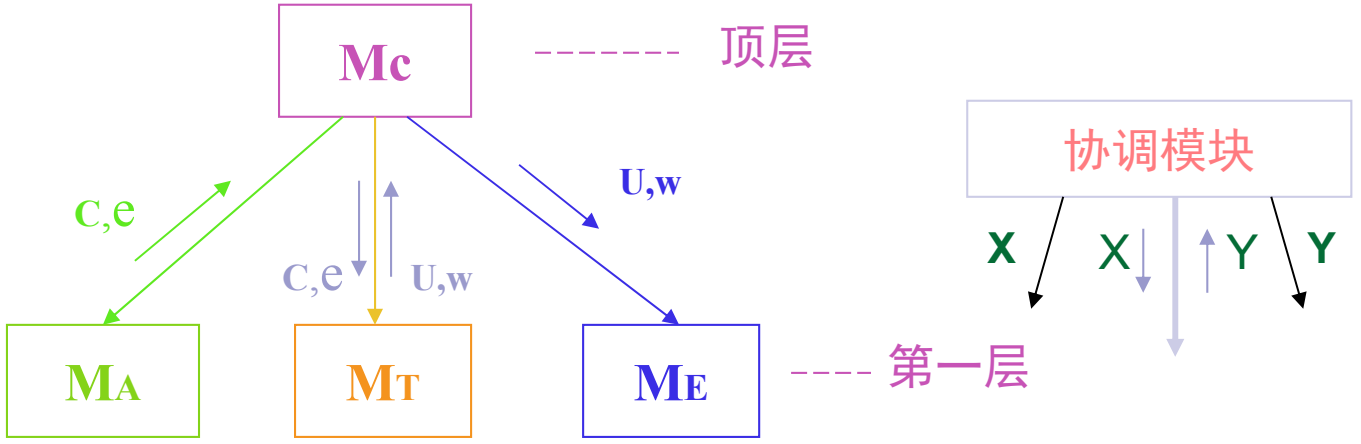
(1)在 DFD 图上标出逻辑输入、逻辑输出和变换中心的分界



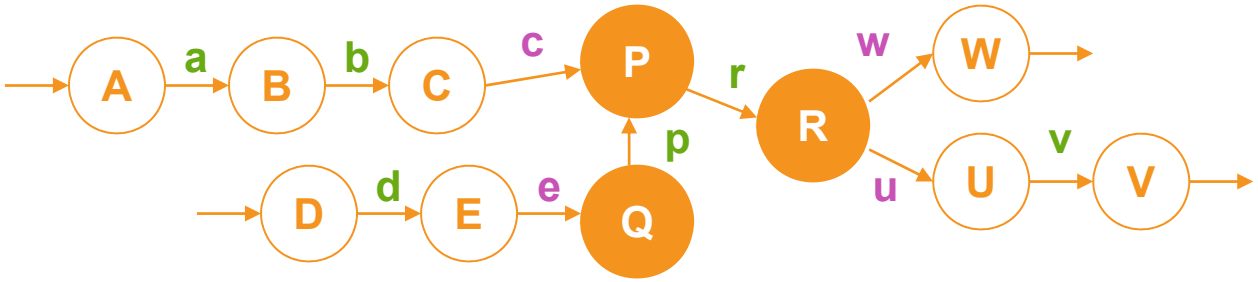
-------- 具有变换型数据流图

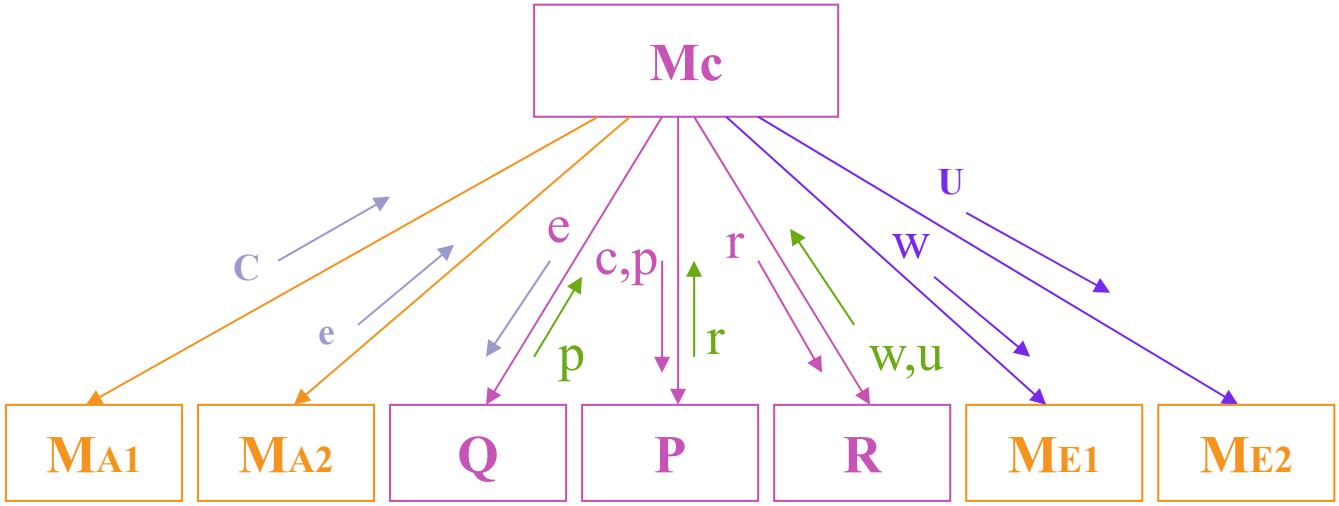
(2) 完成第第一级分解





第一级分解后的 SC 图

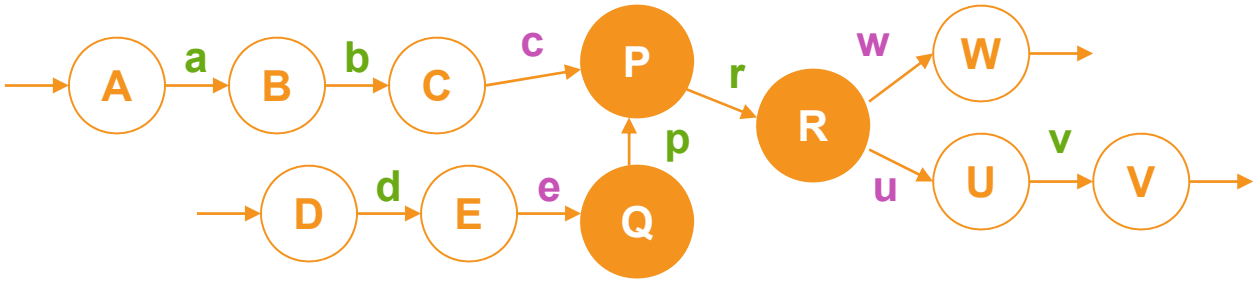


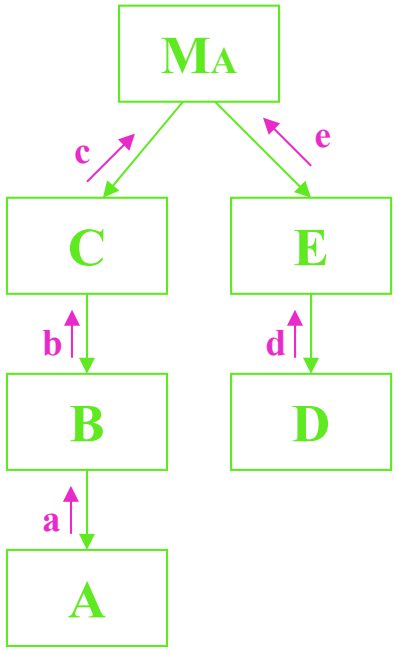


第一级分解后的 SC 图(另一种画法)

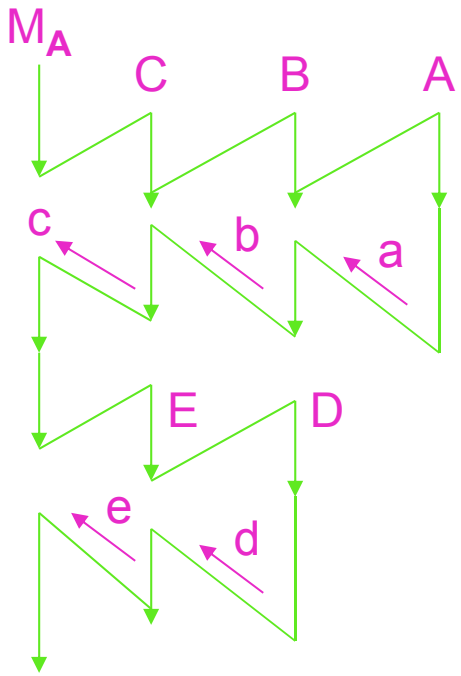
(3) 完成第第二级分解

变换中心

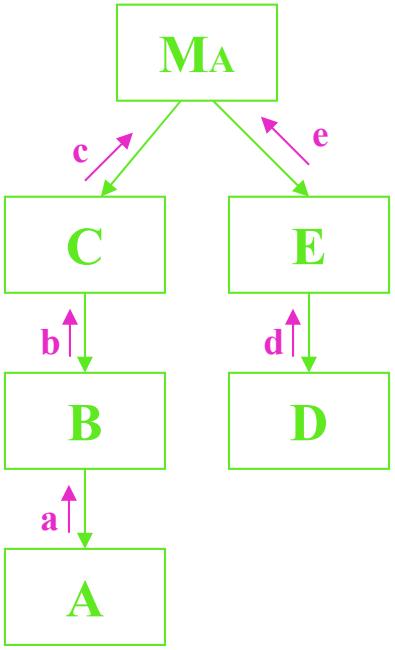


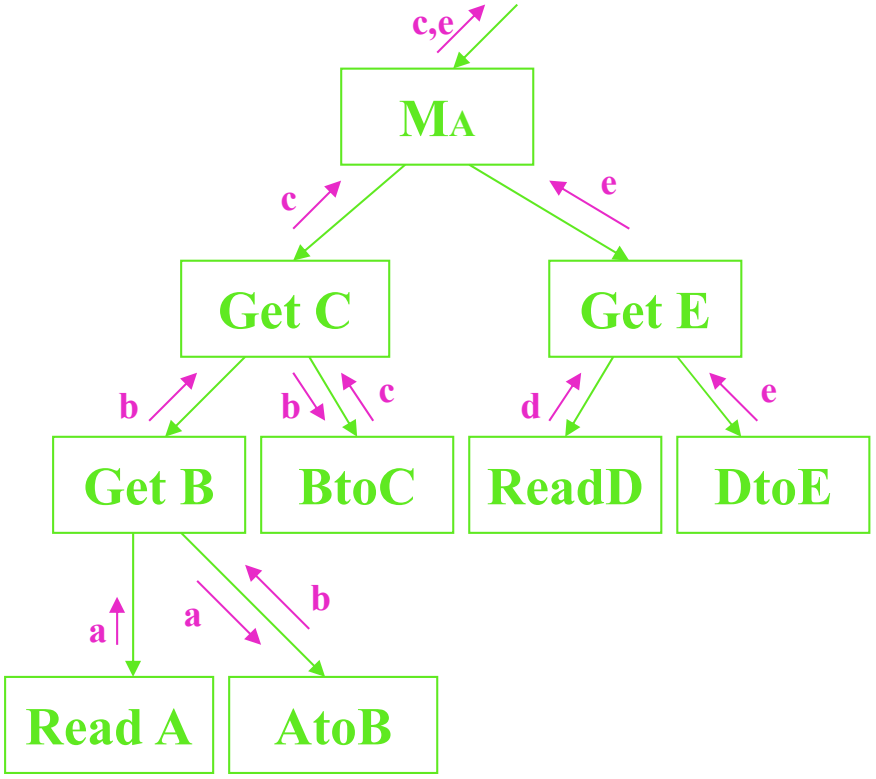


对逻辑输入的分解

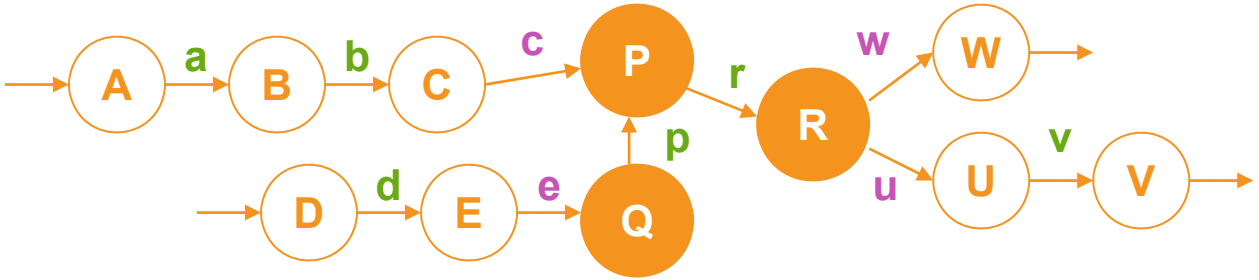


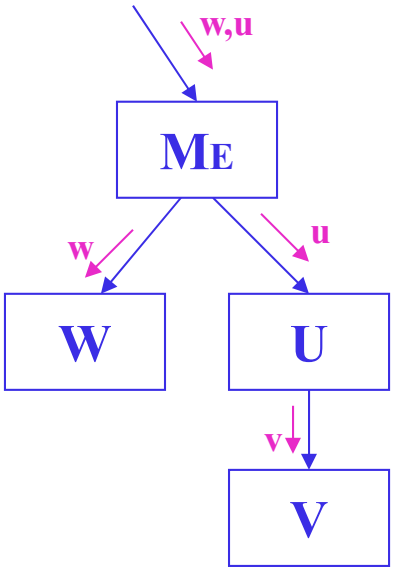
逻辑输入模块的调用与执行过程


MA
c
e
C
E
b
d
B
D
a
A

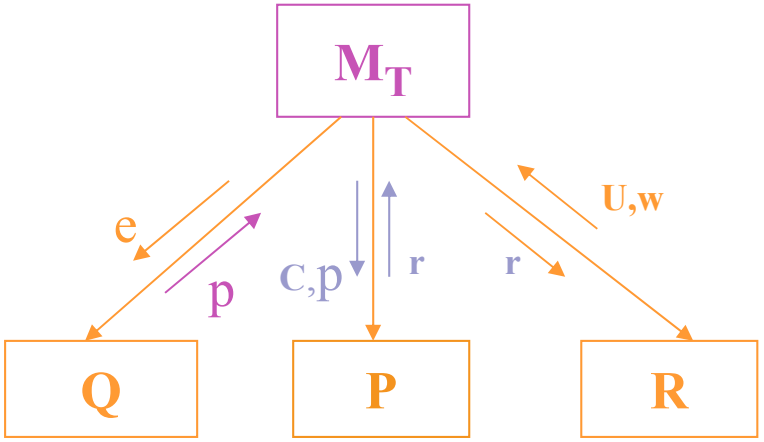

c,e
MA
c
e
Get C
Get E
b
b
c
d
e
Get B
BtoC
ReadD
DtoE
a
a
b
Read A
AtoB

变换中心



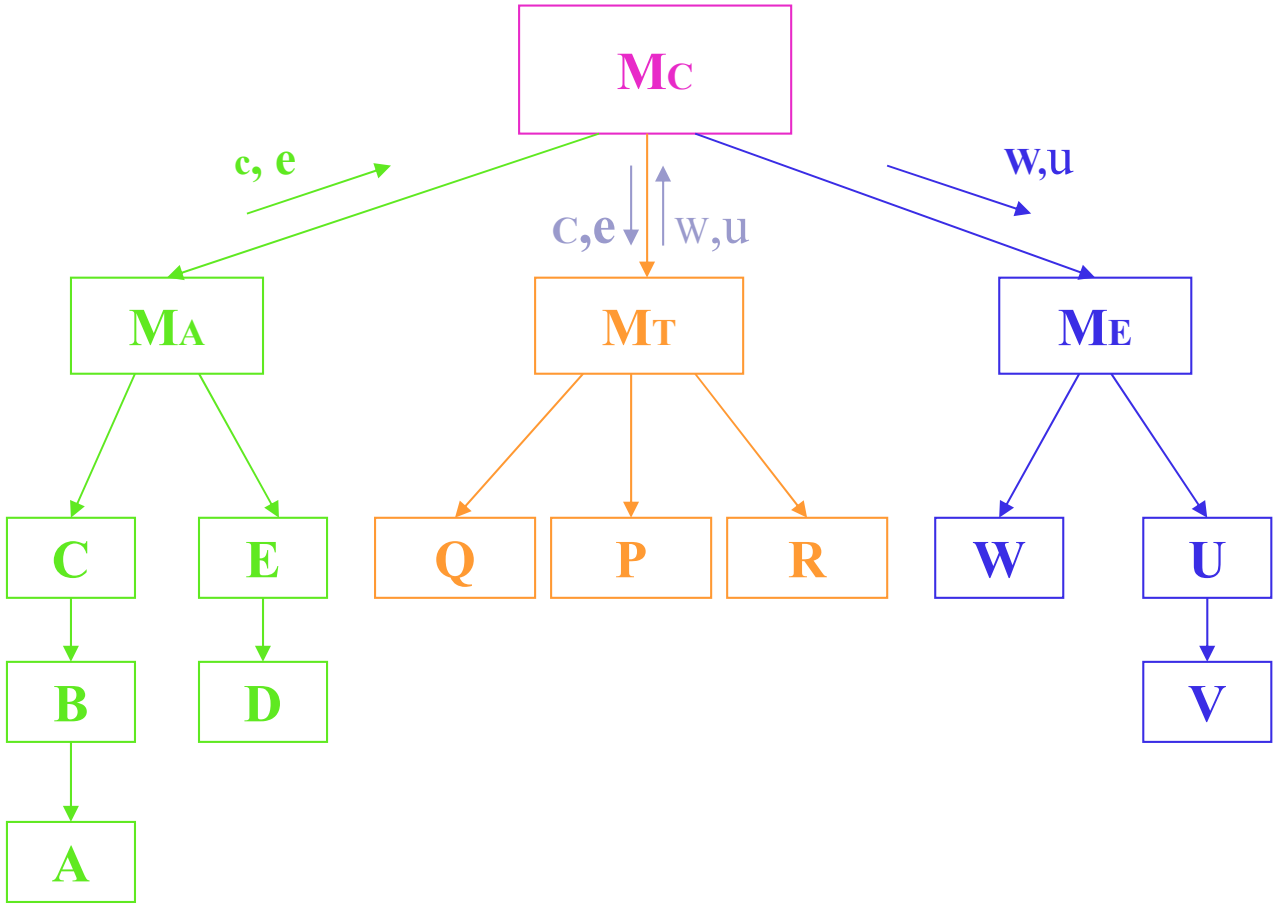


对输出的分解



对变换中心加工的分解

(4) 获得完整的 SC 图



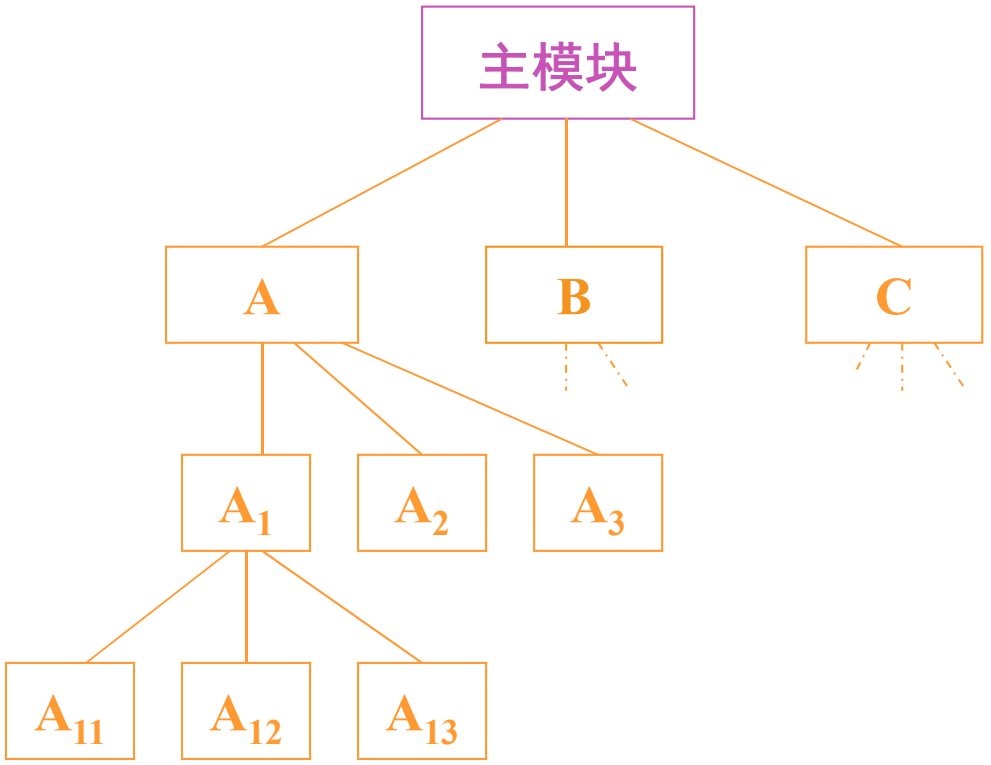
从变换分析导出的初始 SC 图

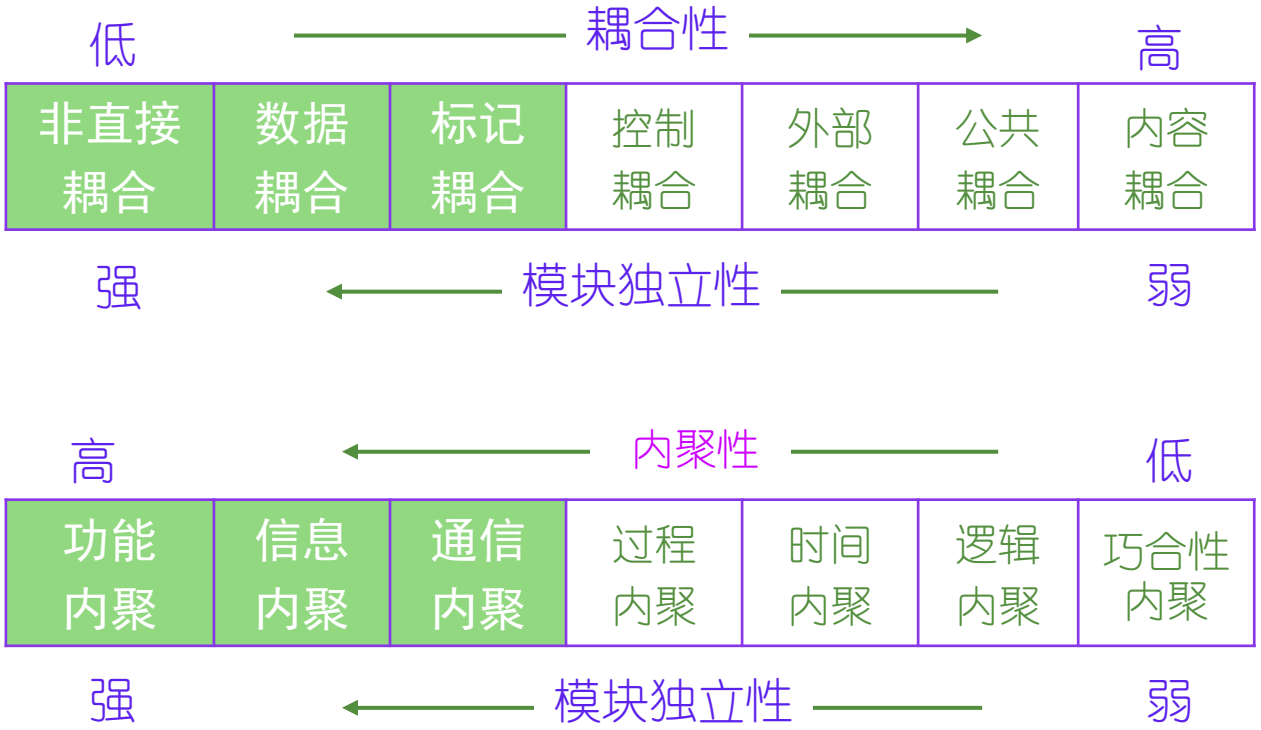
运用变换分析方法建立系统的SC时需注意以下几点：

模块设计的次序时，应遵循对一个模块的全部直接下属模块都设计完成后，再转向另一个模块的下层模块的设计。

在设计下层模块时，应考虑模块的耦合和内聚问题，以提高设计初始**SC**图的质量。

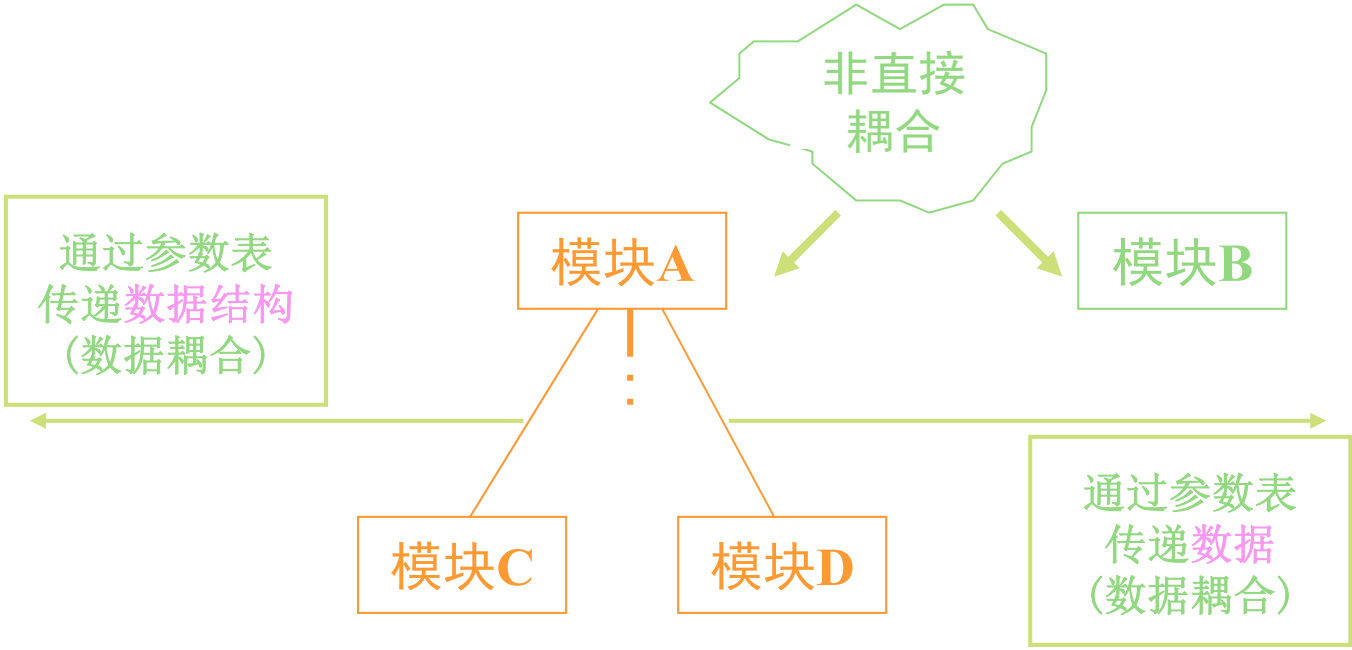
注意“黑盒”技术的使用。


主模块
A
B
C
A_1
A_2
A_3
A_{11}
A_{12}
A_{13}



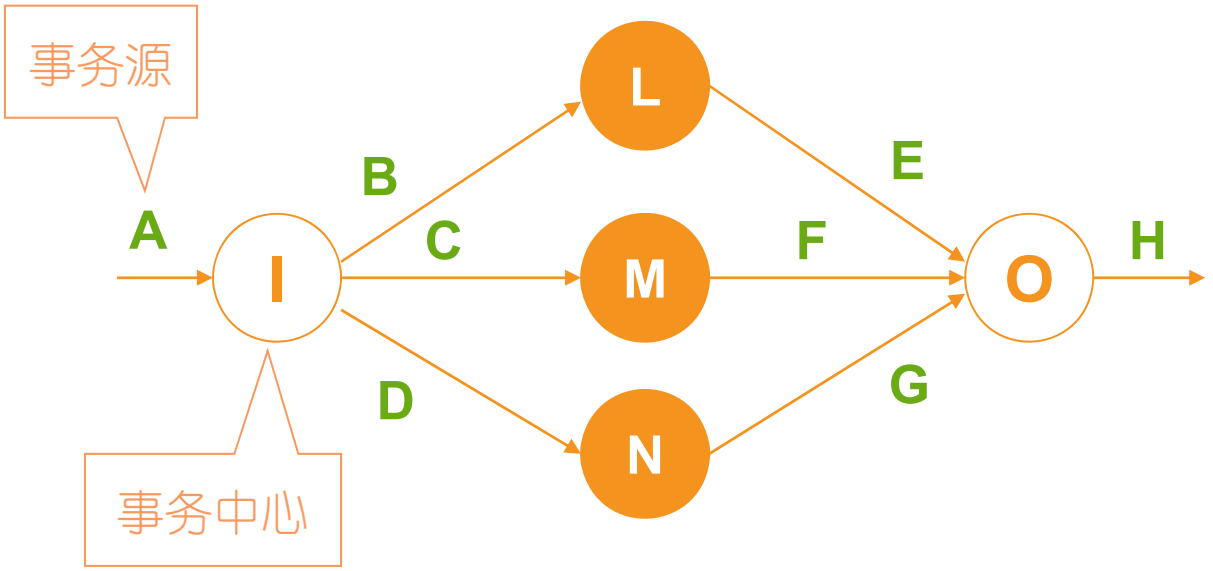
具有高内聚低耦合的模块
才是模块独立性比较强的模块。

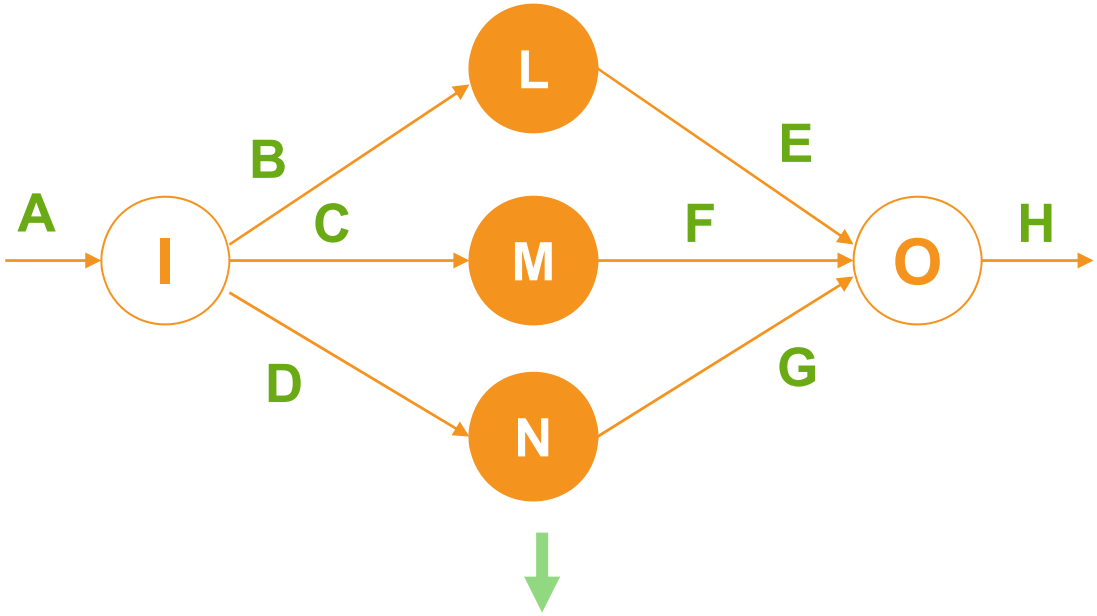
具有高内聚低耦合的模块
才是模块独立性比较强的模块。

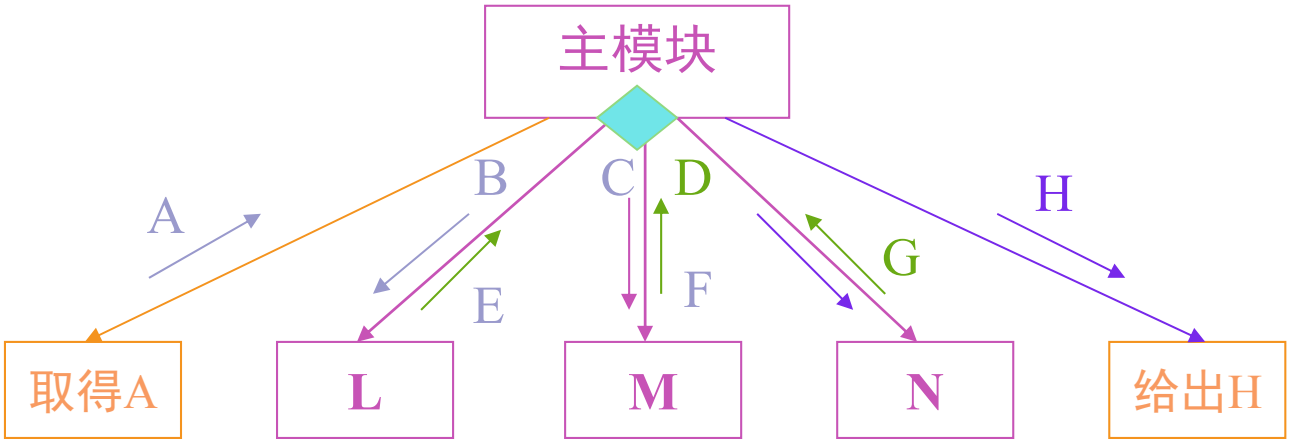


具有松散型 的耦合类型

2、事务分析 ----是将具有事务型的DFD图导出SC图



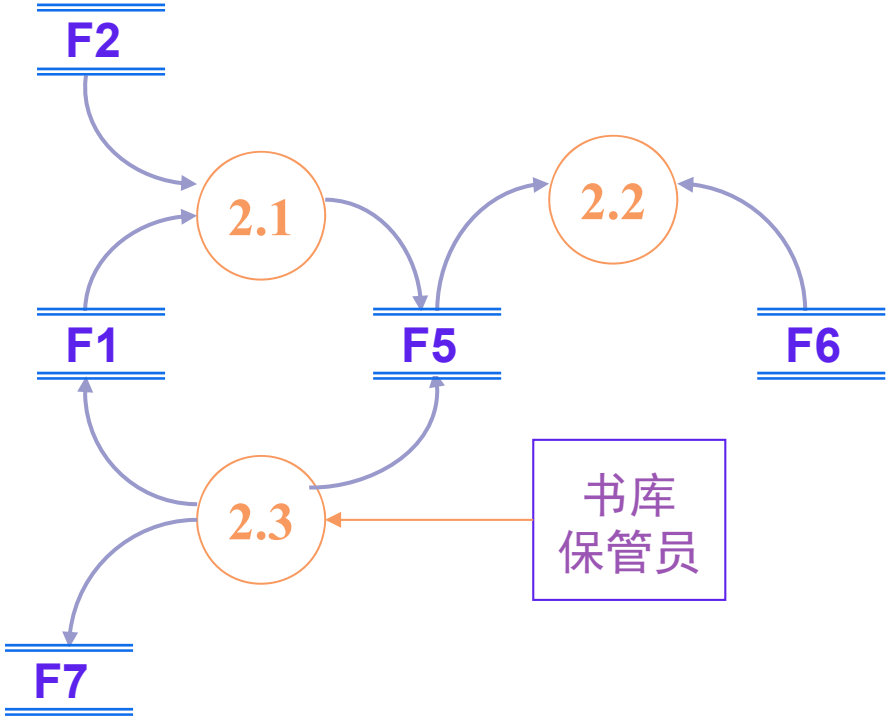

L
B
E
A
C
M
F
O
H
I
D
N
G


主模块
A
B
C
D
H
G
E
F
取得A
L
M
N
给出H

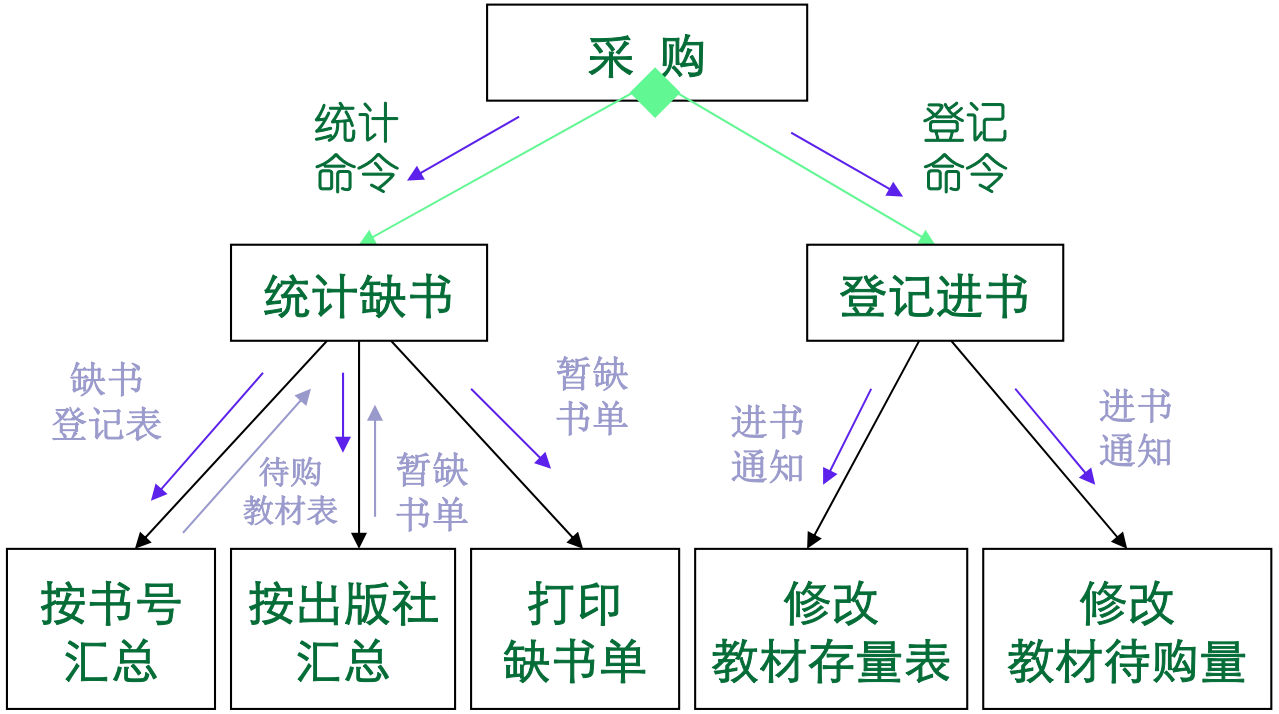
采购子系统 DFD 图转换成SC图

加工名称：
2.1 按书号汇总缺书
2.2 按出版社汇总缺书
2.3 修改教材库存和待购量

文件名称：
F1 教材存量表
F2 缺书登记表
F5 待购教材表
F6 教材一览表
F7 进书登记表

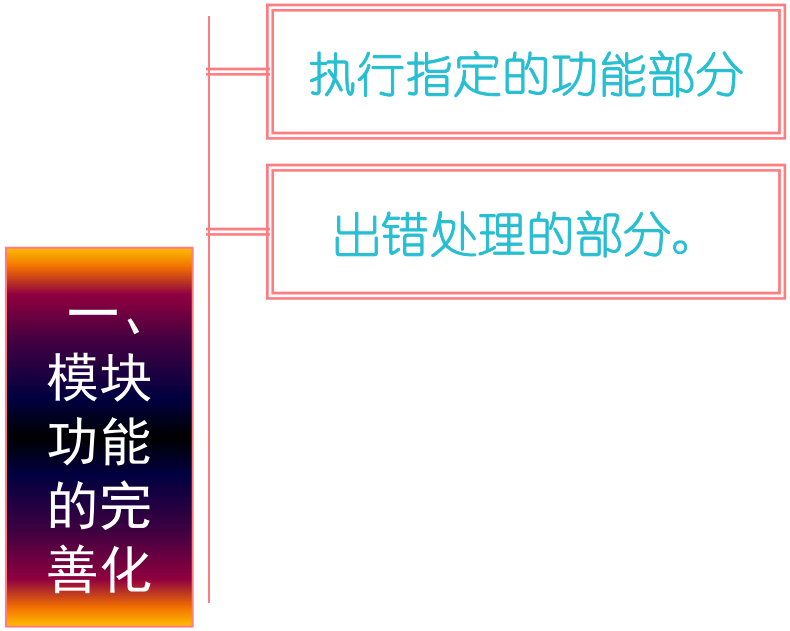


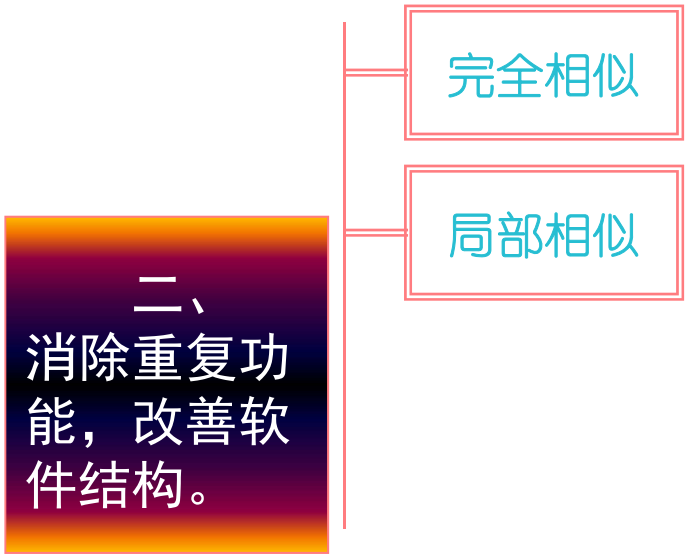
缺书登记表={班号+姓名+书号+数量}



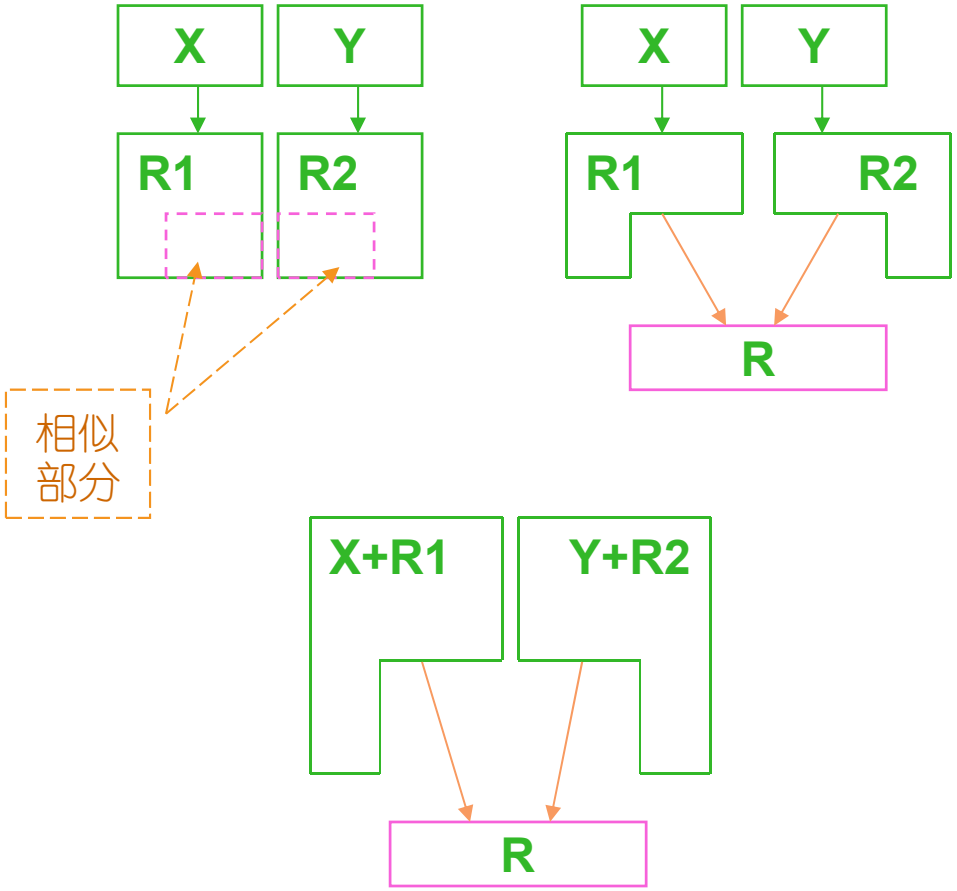
------ 采购子系统的 SC 图 -------

三、软件模块结构的改进




二、
消除重复功能，改善软件结构。
完全相似
局部相似

相似模块的各种合并方案的示意图

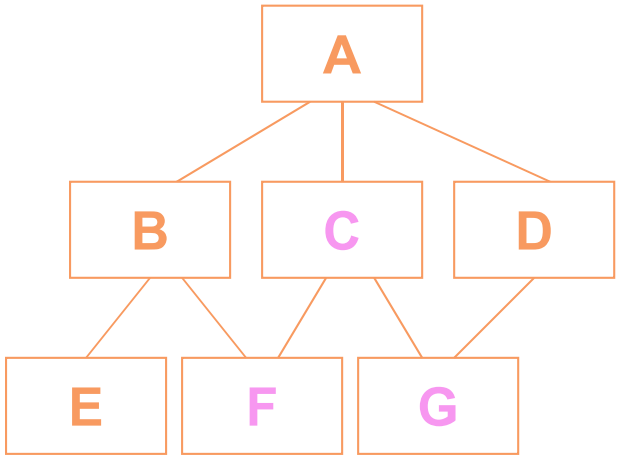


三、
模块的作用范围应在控制范围之内。

控制范围：包括模块本身及其所有的从属模块(即供它调用的模块)。

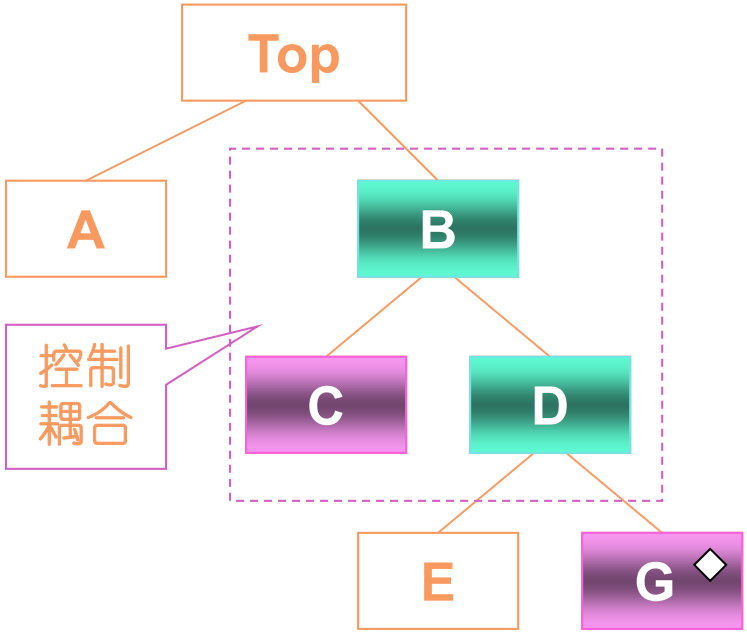
作用范围：是一个与条件判定相关联的所有模块。

模块的控制范围：包括模块本身及其所有的从属模块（即供它调用的模块）。



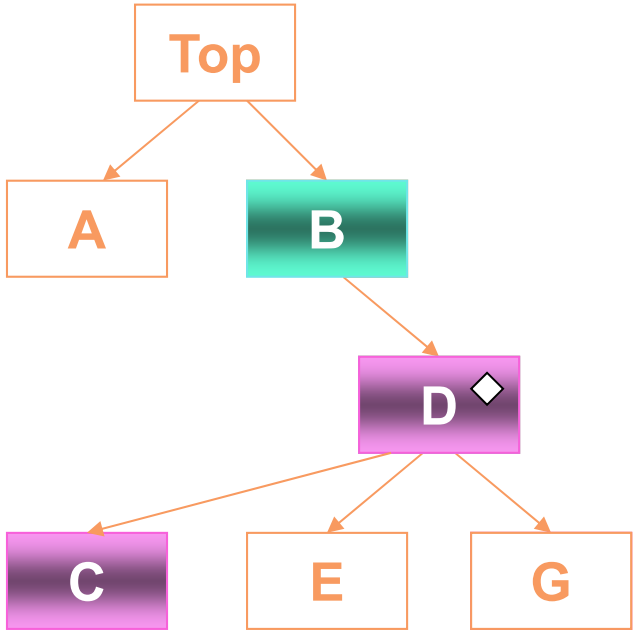
关于模块的控制范围示意图

一个模块的作用范围，是指受这个模块中的判定所影响的模块 。



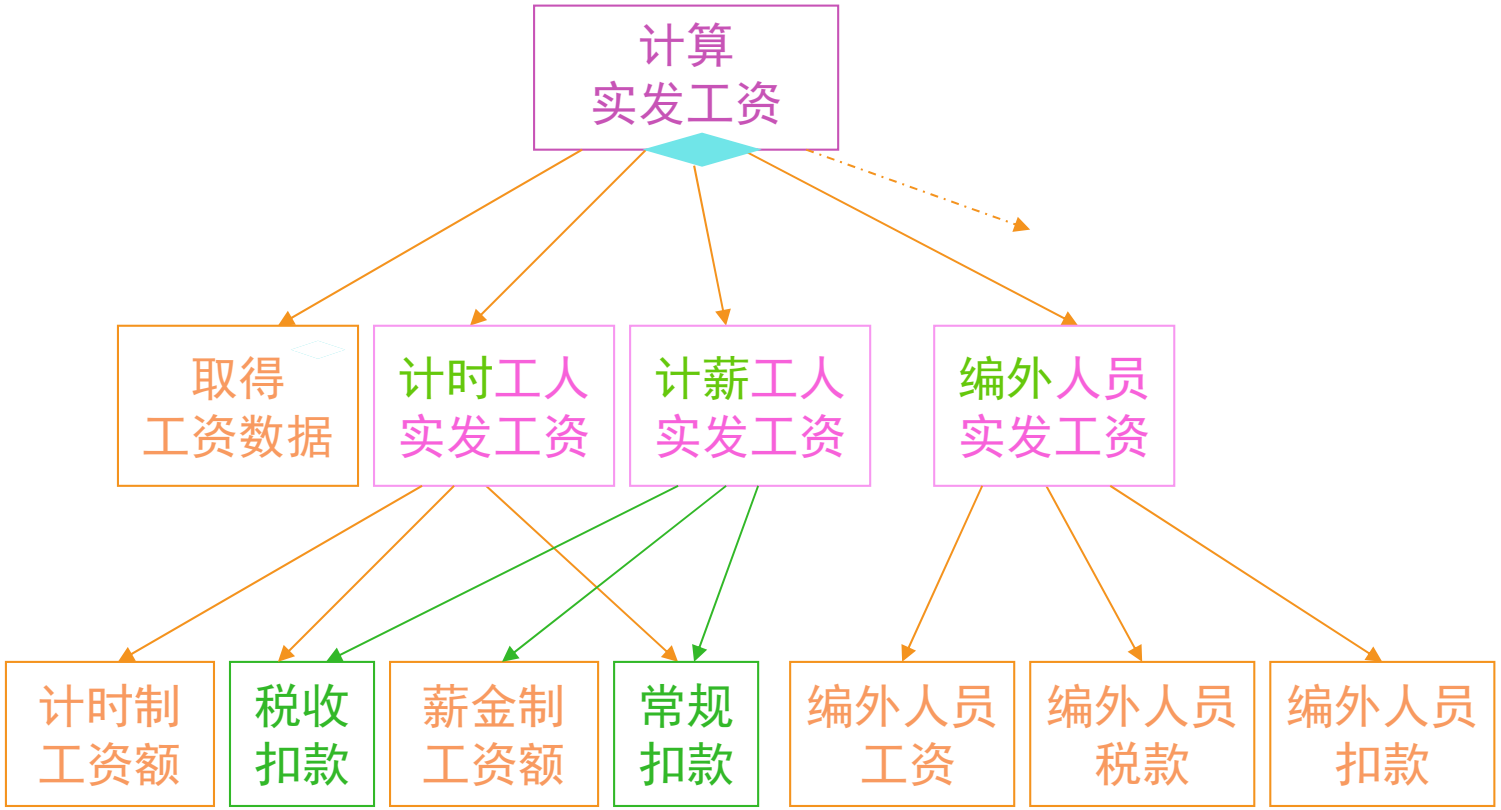
关于模块的作用范围/控制范围的关系示意图

理想的情况，应该使判定的作用范围和判定所在模块的控制范围尽可能地吻合(即应使模块的作用范围尽可能地在控制范围之内)。



对于一个理想 SC图中的模块设计，所有受到一个判定影响的模块应该都从属该判定所在的模块，最好位于作出判定的那个模块本身及它的直接下属模块。

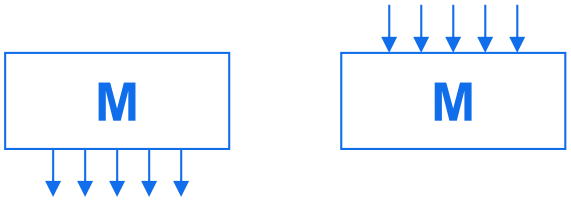
符合作用范围/控制范围的理想判定位置

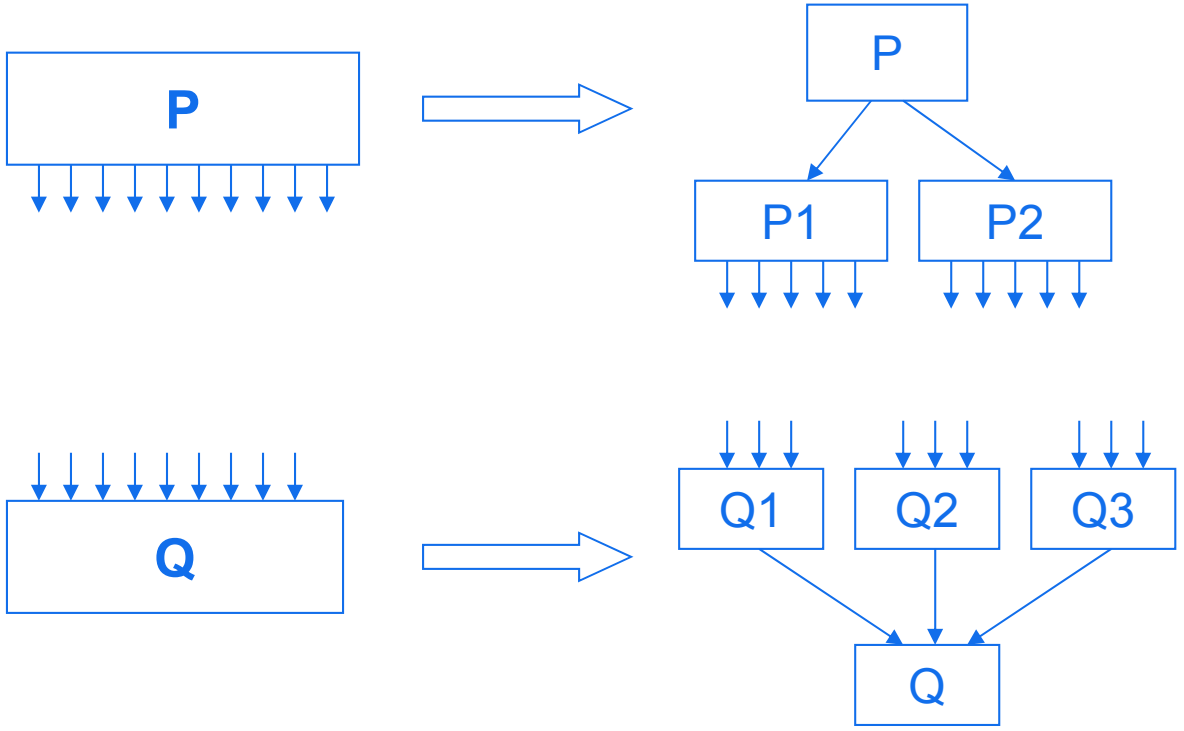

计算
实发工资
取得
工资数据
计时工人
实发工资
计薪工人
实发工资
编外人员
实发工资
计时制
工资额
税收
扣款
薪金制
工资额
常规
扣款
编外人员
工资
编外人员
税款
编外人员
扣款

扇出：指模块调用其下属模块数目。
调用的下属模块数应控制在小于3-4个模块。

扇入：指模块的上级模块数。（即共有多少个模块需要调用这个模块）

四、
尽可能地减少高扇出结构，随着深度增大扇入。

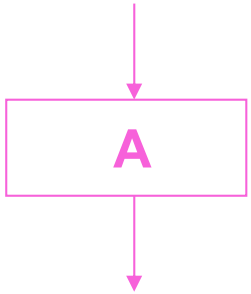



P
P
P1
P2
Q
Q1
Q2
Q3
Q

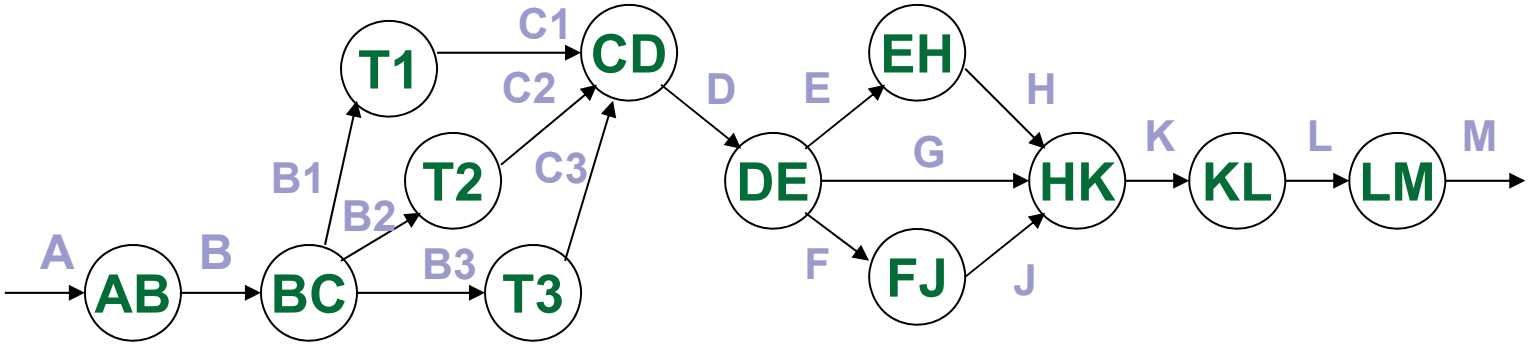
模块的大小，可以用模块中所含语句的数量的多少来衡量。 50-100

五、
模块的大小要适中。

六、应设计出功能可预测的模块，但要避免过分受限制的模块。



请将下列给出的 DFD 图转换成SC图

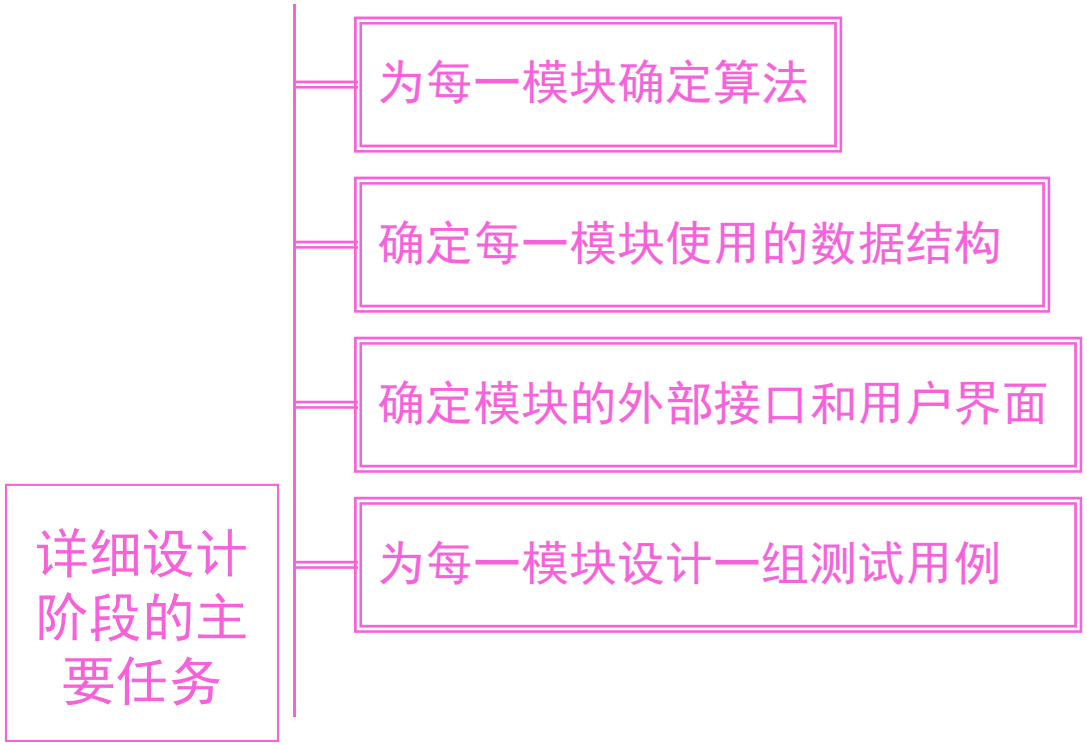


4.4 详细设计描述的工具

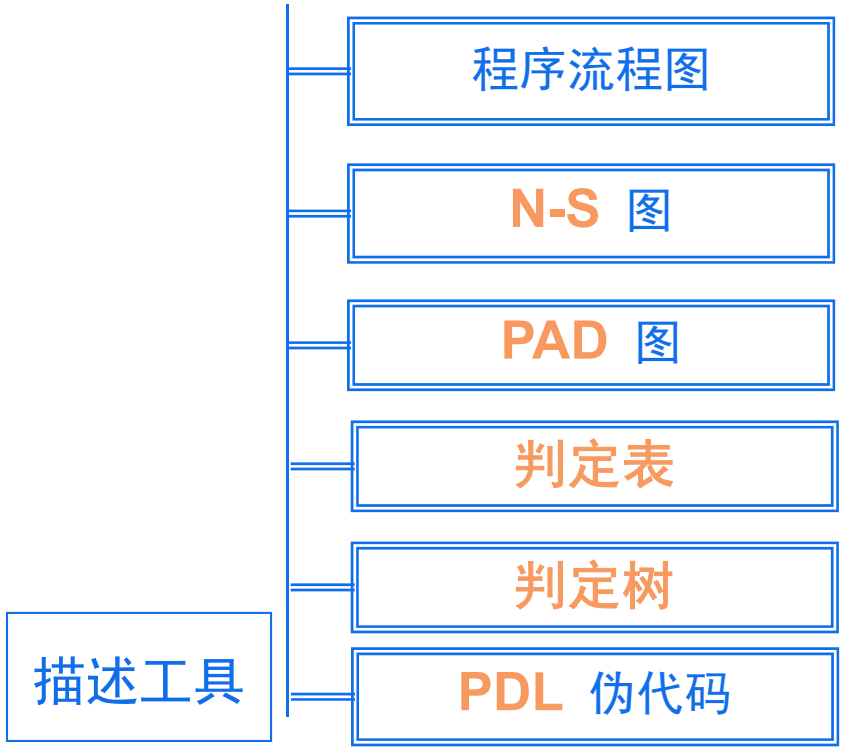
5.1 详细设计阶段的目的与任务

详细设计的目的：为软件结构图（SC）中的每一个模块确定采用的算法和模块内数据结构，用某种选定的表达工具给出清晰的描述。

详细设计阶段的主要任务：编写软件的 “详细设计说明书”


详细设计阶段的主要任务
为每一模块确定算法
确定每一模块使用的数据结构
确定模块的外部接口和用户界面
为每一模块设计一组测试用例

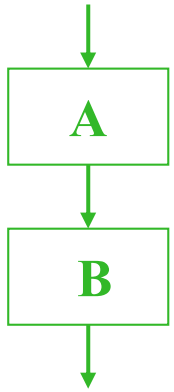
5.2 详细设计阶段的描述工具



一、程序流程图

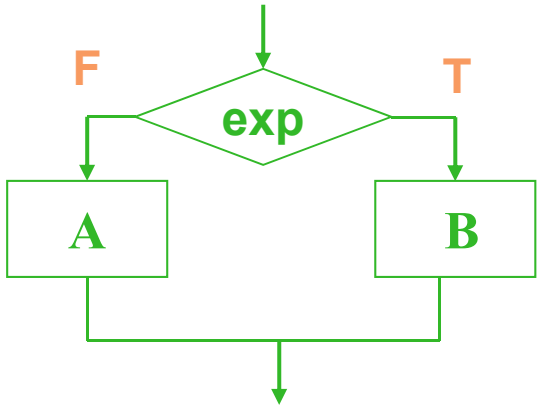
1、顺序型

几个连续的加工依次序排列



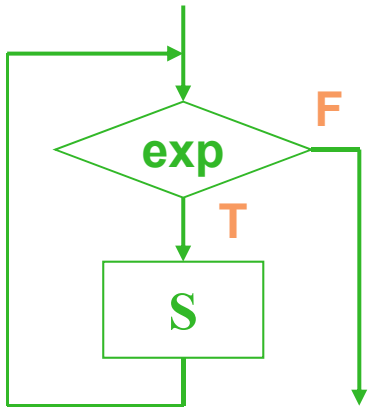
2、选择型

由某个判断式的取值决定选择两个加工中的一个。



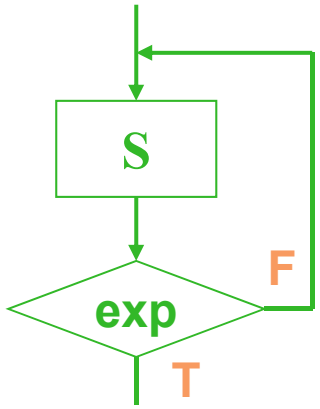
3、当型循环型

当循环控制条件成立时，重复执行特定的加工。



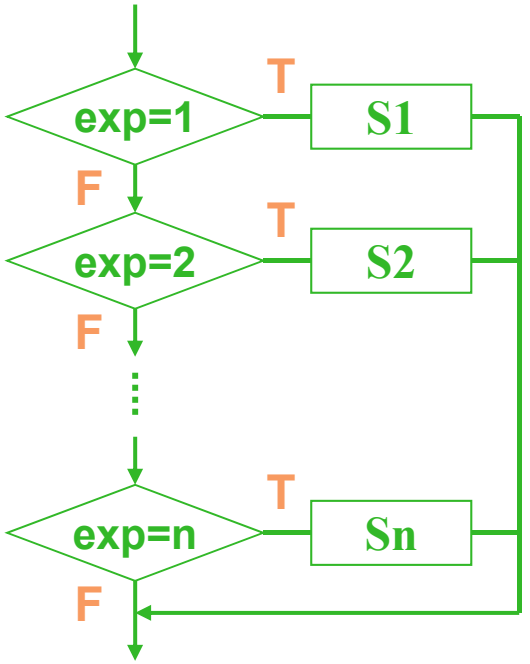
4、直到型循环型

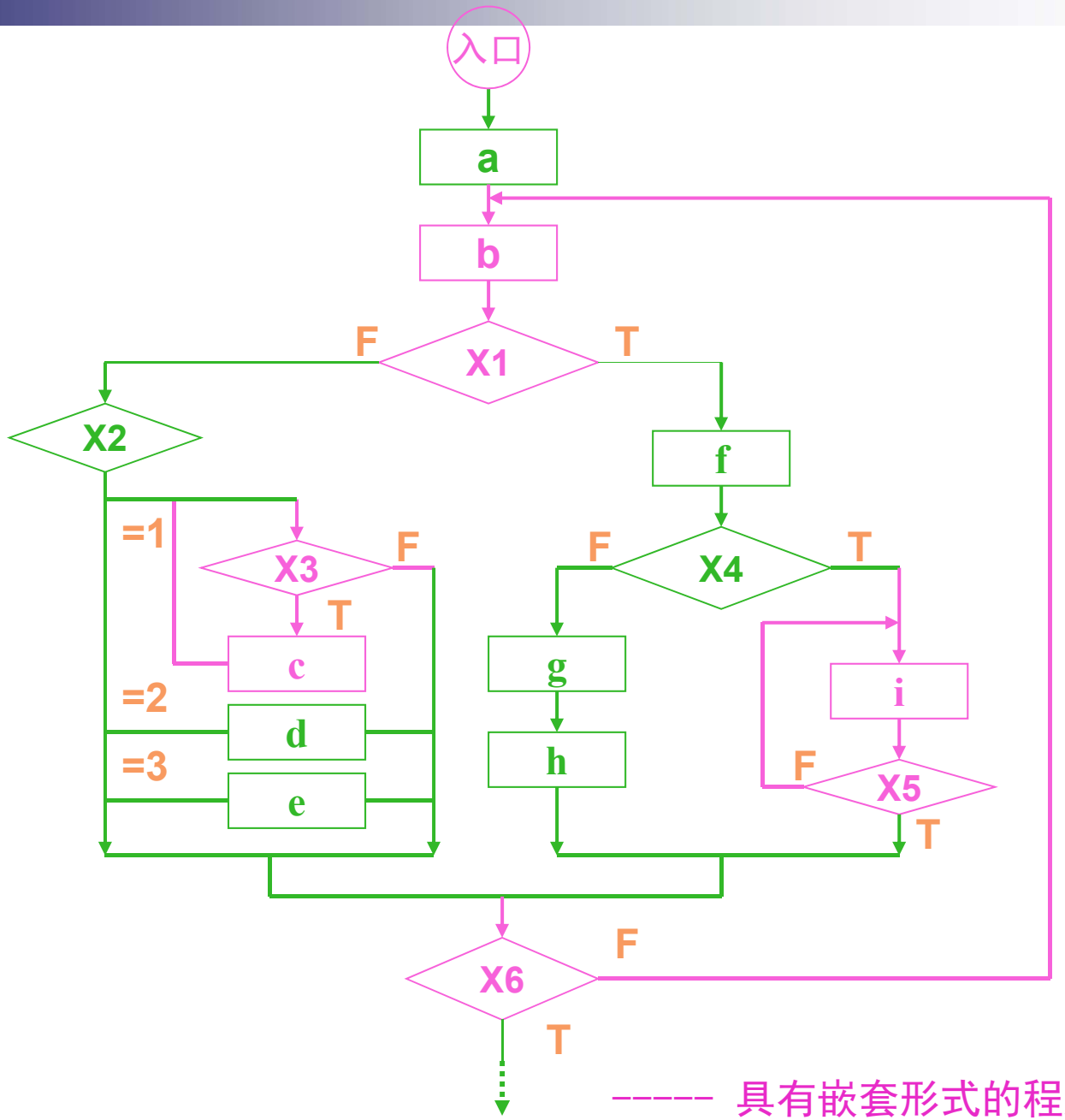
重复执行特定的加工，直到循环控制条件成立时。



5、多情况选择型

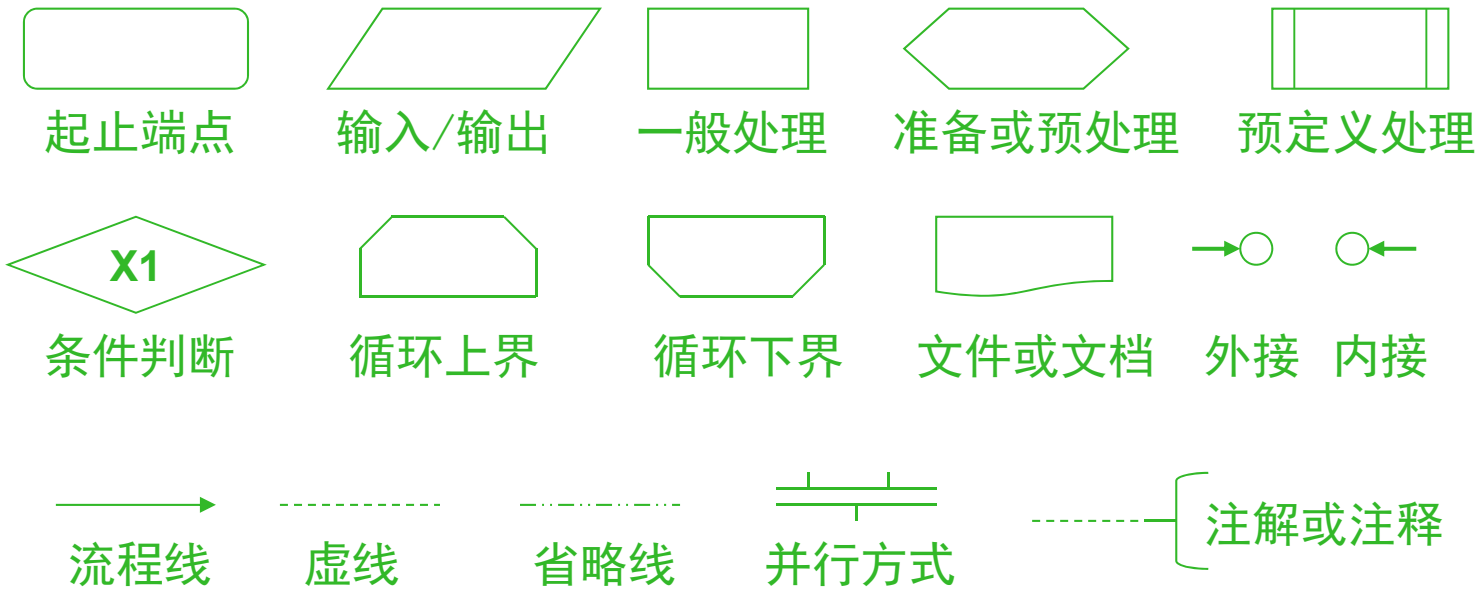
列出多种加工情况，根据控制变量的取值，选择执行其一。





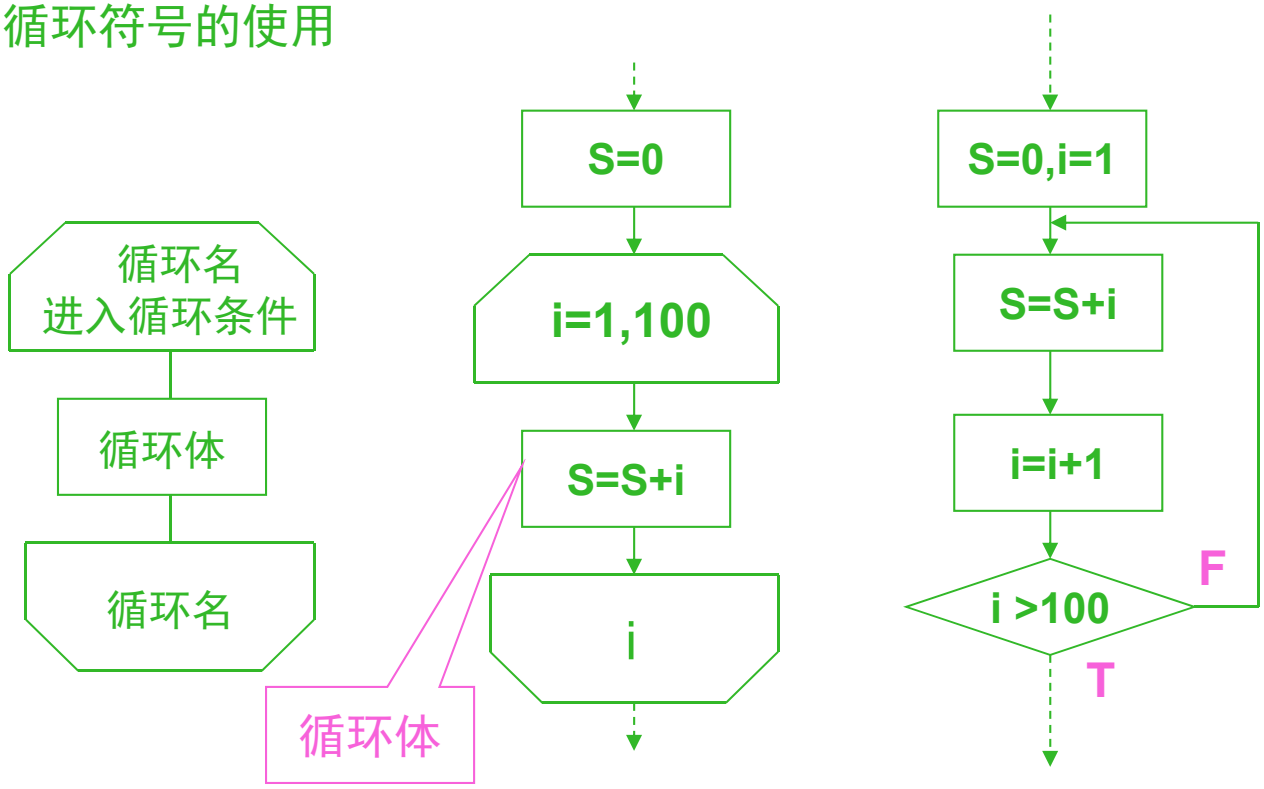
----- 具有嵌套形式的程序流程图

标准化程序流程图规定符号



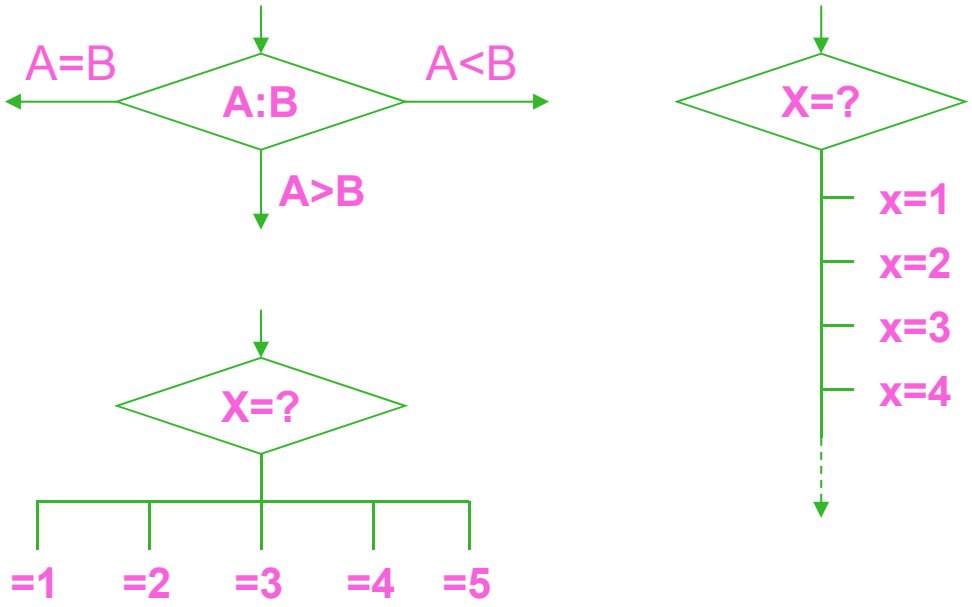
流程符号的使用规则

1、循环符号的使用



------ 循环流程符号的使用

2、判断有一个入口，但也允许有多个可选出口



------ 多出口判断流程符号的使用

练习

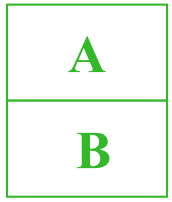
请利用程序流程图描述下列问题的程序结构

某汽车修配厂，有一个存有汽车零件的仓库，其中存有若干种零件，请编写一个查询程序，用于查询该库中某零件的库存量为多少。

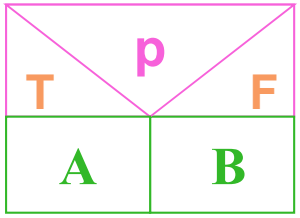
设该模块为 查询模块。请设计该模块的程序结构。具体要求：

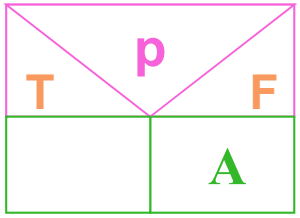
1、应具有重复查询功能；
2、应具有数据检测功能；
3、请利用程序流程图描述该模块的算法。

二、N-S 图 ------- Nassi and Shneideman

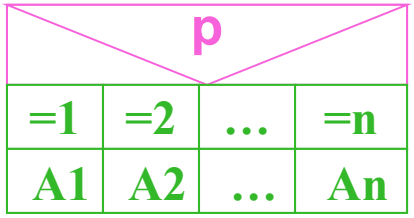


顺序型

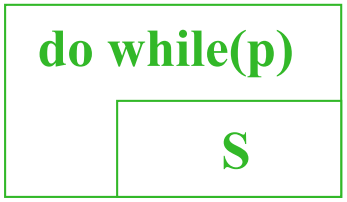




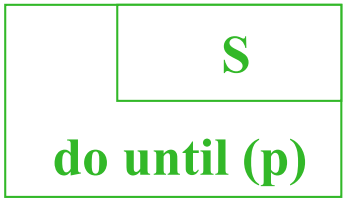
选择型



多分支选择型

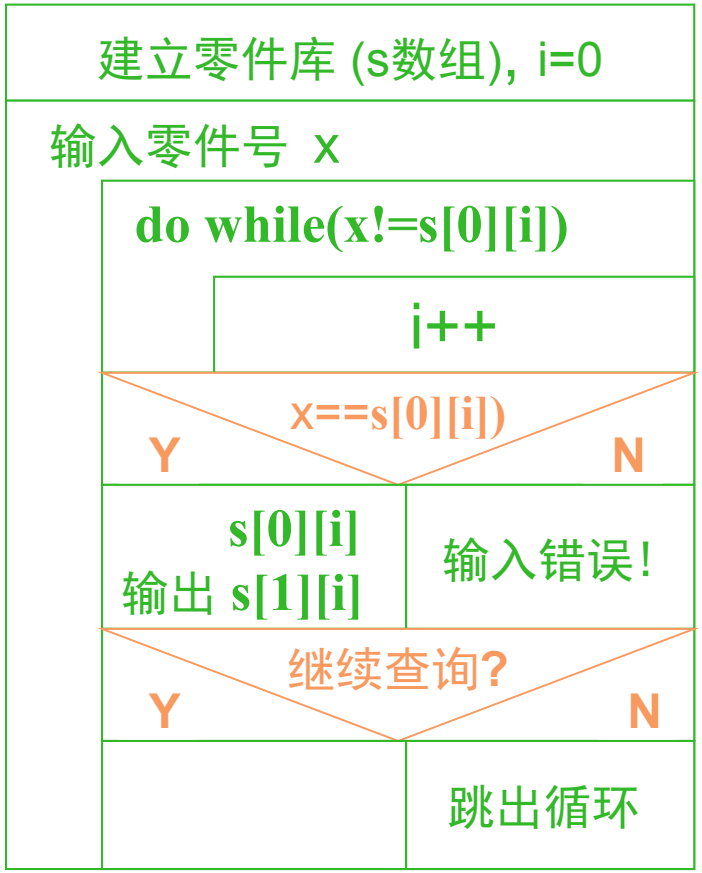


当型循环型

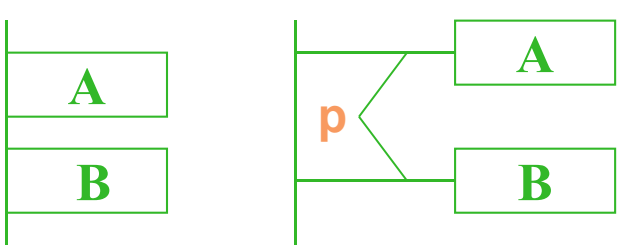


直到型循环型

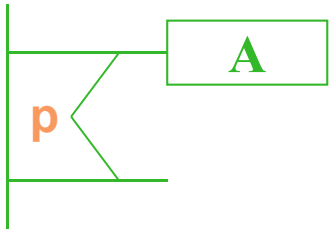
零件号	10001	10002	10003	10004	10005	10006
库存量	1000	1250	886	69	2020	3450



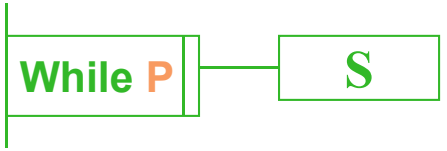
三、PAD 图 ------- Problem Analysis Diagram



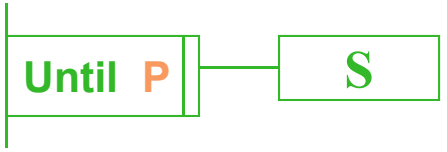
顺序型



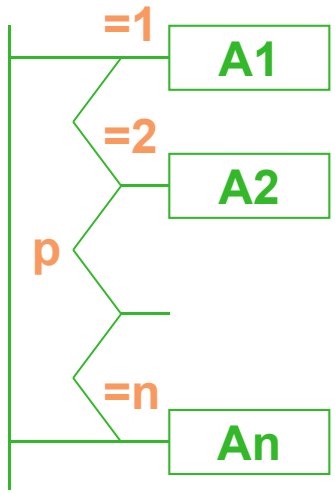
选择型



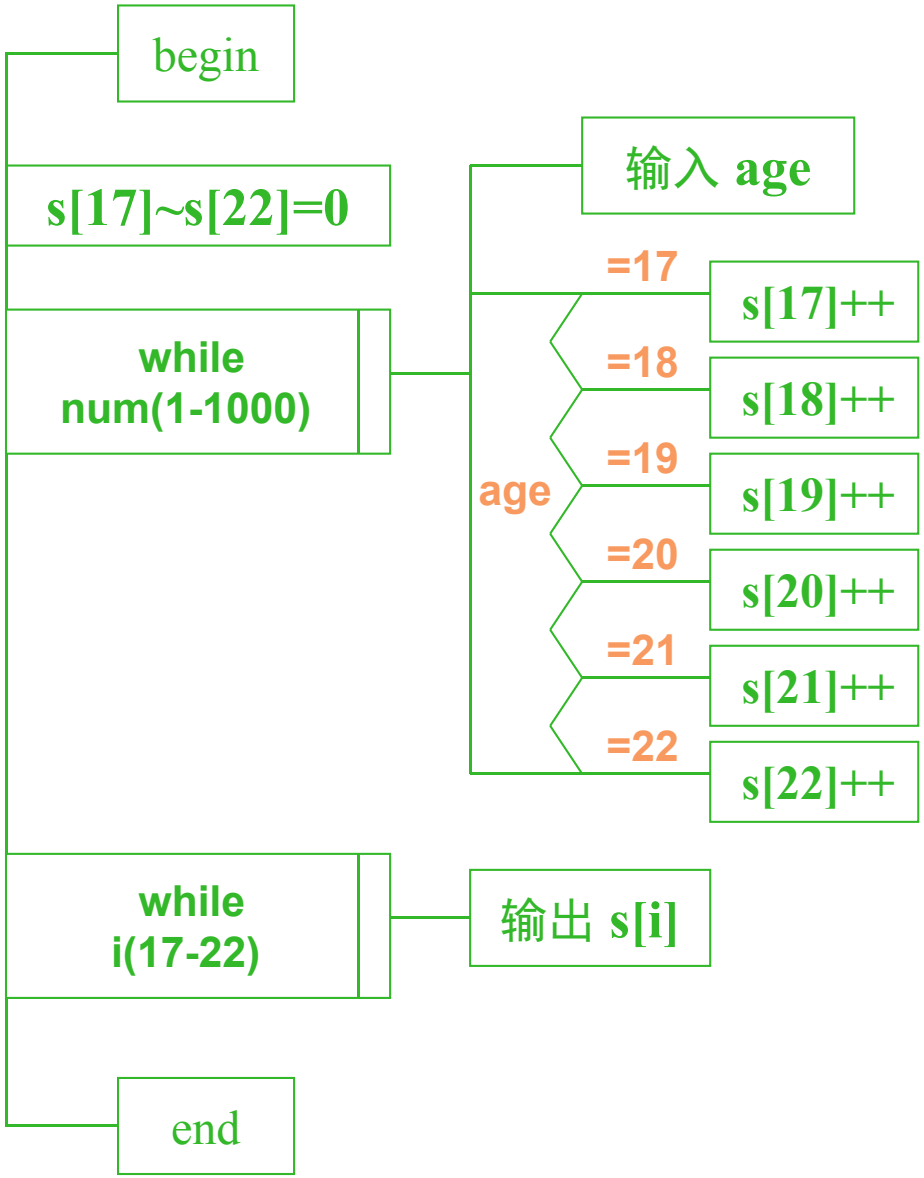
当型循环型



直到型循环型



多分支选择型循环型

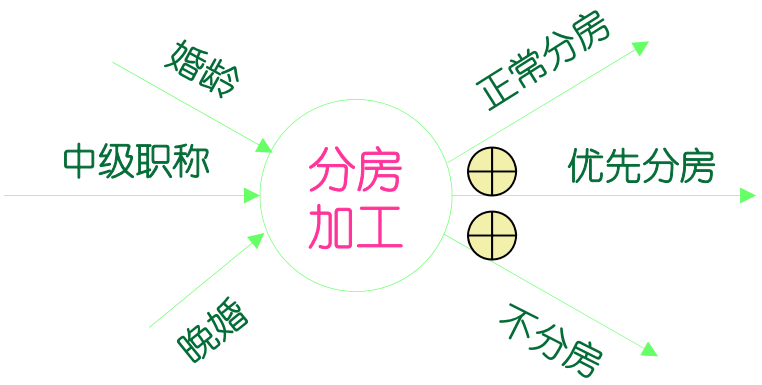

begin
s[17]~s[22]=0
while
num(1-1000)
输入 age
age
=17
s[17]++
=18
s[18]++
=19
s[19]++
=20
s[20]++
=21
s[21]++
=22
s[22]++
while
i(17-22)
输出 s[i]
end

举例

四、判定表

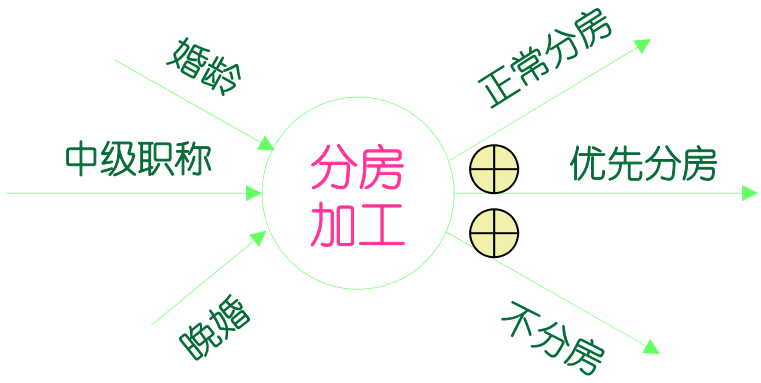
判断表

采用表格的形式来表达
具复杂判断的加工逻辑



1、一般职工婚后5年可参加分房
2、中级以上职称的职工婚后3年可参加分房
3、符合正常分房条件的职工，若再符合晚婚条件可优先分房

分房加工逻辑

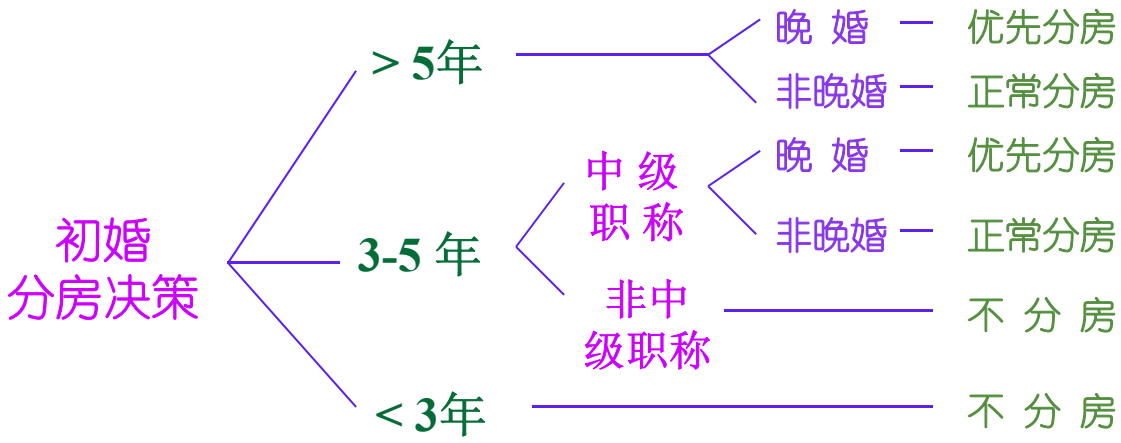


		1	2	3	4	5	6	说　明
条件	婚　龄	>5年		3-5年			<3年	(-) 表示任意 (Y) 条件满足 (N) 条件不满足 (*) 选中的决策
	中级职称	----------		Y		N	-----	
	晚　婚	Y	N	Y	N	----	-----	
决策	优先分房	*		*				
	正常分房		*		*			
	不分房					*	*	

五、判定树

判断树

判断树是判断表的图形形式
其适用场合与判断表相同



“检查发货单” 的判断表

		1	2	3	4
条件	发货单金额	>$500	>$500	<=$500	<=$500
	赊欠情况	>60天	<=60天	>60天	<=60天
决策	不发出批准书	V			
	发出批准书		V	V	V
	发出发货单		V	V	V
	发出赊欠报告			V	

把下列用文字叙述的内容请采判断表和判断树描述出来

设某旅游票预定系统中，在旅游旺季 7- 9、12月份，如果订票超过 50张，则优惠票价的 15% ；50张以下，优惠5%。在旅游淡季1-6，10、11月份，若订票超过 50张，则优惠30%；50张以下，优惠 20%。

六、PDL ------- Program Ddesign Language

PDL 是一种用于描述功能模块的算法设计和加工细节的语言。称为设计程序用语言。它是一种伪代码(Pseudo code)

PDL -----关键词+自然语言

（1） 数据说明:　其功能是定义数据的类型和作用域

格式: TYPE <变量名> AS <限定词1> <限定词2>

说明: 1. 变量名:是一个模块内部使用的变量或模块间共用的全局变量名。

2. 限定词1 :标明数据类型

3. 限定词2 :标明该变量的作用域

TYPE number AS STRING LENGTH (12)

（2） 程序块:　PDL的过程成分是由块结构构成的，而块将作为一个单个的实体来执行。

```
BEGIN <块名>
        < 一组伪代码语句>
END
```

(3) 子程序结构: 把 PDL 中的过程称为子程序。

```
PROCEDURE <子程序名> <一组属性>
  INTERFACE  <参数表>
          < 程序块或一组伪代码语句>
END
```

(4) 基本控制结构:

```
IF <条件>
  THEN  <程序块/伪代码语句组>;
  ELSE   <程序块/伪代码语句组>;
ENDIF
```

--- 选择型结构

```
DO WHILE <条件描述>
    <程序块/伪代码语句组>;
ENDDO
```

```
REPEAT UNTIL <条件描述>
    <程序块/伪代码语句组>;
ENDREP
```

--- 重复型结构

```
DO LOOP <条件描述>
    <程序块/伪代码语句组>;
    EXIT WHEN
ENDLOOP
```

```
DO FOR <下标=下标表，表达式>
    <程序块/伪代码语句组>;
ENDFOR
```

--- 重复型结构

```
CASE OF <case 变量名>;
    WHEN < case 条件1> SELECT <程序块/伪代码语句组>;
    WHEN < case 条件2> SELECT <程序块/伪代码语句组>;
    … …
    DEFAULT: < 缺省或错误case:  <程序块/伪代码语句组>;

ENDCASE
```

------ 多路选择结构

READ/WRITE TO <设备> <I/O表>

--- 输入/输出结构

```
Enter a vector
Set Maximum to the value of the first
      element in the vector
DO for each second one to the last
      IF value of THEN element is greater
         than the Maximum value
      Set Maximum to value of the element
ENDDO
Print the Maximum value
```

```
Input array A
Max=A(1)
DO for I=2 to N
   IF Max<A(I)
      Set Max=A(I)
   ENDIF
ENDDO
Print Max
```

练习

请按下列给出的文字要求，用 PDL 描述其该模块的算法

设某模块的功能是：读入任意长的一段英文课文，将其分解为单字。然后输出一个单词表，并指出每个单词在课文中所出现的次数。

请将下列的 PDL 表示的某模块的过程性描述，
改为用：1、N-S 图　2、PAD 图表示

```
… …
execute process a
REPEAT UNTIL condition X8
    execute process b
    IF condition X1
        THEN BEGIN
                execute process f
                IF condition X6
                    THEN REPEAT UNTIL condition X7
                            execute process i
                         ENDREP
                    ELSE BEGIN
                            execute process g
                            execute process h
                         END
                ENDIF
             END
```

```
        ELSE CASE OF Xi
                WHEN condition X2 SELECT
                   DO WHILE condition X5
                        execute process C
                   ENDDO
                WHEN condition X3 SELECT process d
                WHEN condition X4 SELECT process e
             ENDCASE
        ENDIF
  ENDREP
  execute process j
END
```

用户界面设计-做减法

- 我们大家平时都说要向某某大师或某某产品学习，把最重要的功能做好交给用户，把那些无关紧要的功能藏起来，做减法……但是程序员们还是会想着把高级功能“秀”出来。我们都用过各种**电视/DVD播放器的遥控器**，功能很强，按钮很多吧？你有没有注意到老人家使用遥控器时的困难？

姥姥的遥控器

电子邮箱的UX

- 领导/项目经理：要一个电子邮箱地址，让人民群众能发邮件投诉假币和其他事情！
- 技术人员：好，内网地址搞好了，工具自动转成外网的地址。搞定！
- 测试人员：　把邮件地址复制/粘贴到电子邮件的地址栏，发送一个邮件试试看，收到了么？收到了。好，通过！
- 项目经理：好，把邮件地址印成提示牌，搬到各地的营业处去！

实例：


人民币流通和反假货币举报机制投诉电话和投诉电子邮箱
投诉电话为：95566
投诉电子邮箱：0xLJB3F6C4C9D3EBCFD6BDF0B9DCC0E
DzFJPJKFBCNYXJGL@mail.notes.bank-of-china.com
@木遥
weibo.com/farmostwood

替用户着想 – 他们有多蠢？

- 微软必应搜索有一个“实时显示英语解释”的功能，但是这个功能把鼠标所在的所有英语单词都解释一下，包括小学生都懂的“a, of, at, on, and, the, he, she, ...”，用户的鼠标常常会无意地停留在这些词语上面，你就会看到这个“英语翻译”功能自作多情地告诉你“a”是什么意思，顺便把页面的其他文字给遮住了：
- 我们的PM/Dev/Test在设计/实现/测试这个功能的时候想过目标用户的英文水平是什么样的么？他们需要哪个程度的英文解释？如果他们连“a”都不懂，他们能来到你这个网页搜索含有英文的结果么？！
- 参见：http://cn.bing.com

需要解释 “a” 这个单词么

The Inmates Are Running the Asylum : Why High Tech Products ...
book.douban.com/subject/1440449
作者: Alan Cooper · 288 页 · 2004 · 37 条评论
Imagine, at a terrifyingly aggressive rate, everything you regularly use is being equipped with comput ars-everything-being ...

a [ə]
n. 英文字母表的第一字母
na. 一
必应词典 下载客户端 关闭屏幕取词

The Inma
book.douba
作者: Coope
【以下内容算是读书时候一些随想，想到哪写到哪，未必很有逻辑】在做用户访谈或者可用性测试的时候经常会遇到一种情况，就是用户常常为自己在某件产品的使用不便 ...

相关搜索

the end of the world
the gift of the magi
the 比较级 the 比较级
the dock of the bay
death row inmates
43 inmates executed 2011
the cat and the bell
the same as the same that

替用户着想 – 越用越好用

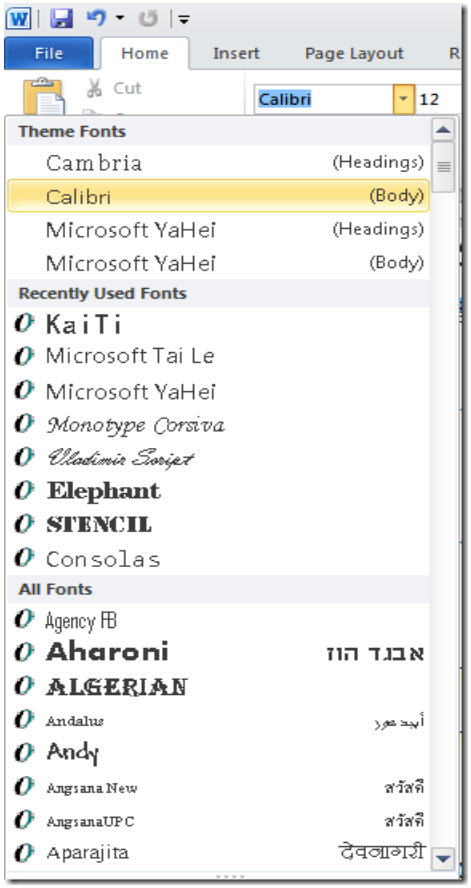
- 但是当用户已经是第N次使用你的产品时，你的UI能否为这些用户提供方便呢？你的产品是下面的哪一种：
- a）软件用得越多，一样难用；
- b）软件用得越多，越发难用；
- c）软件用得越多，越来越好用。

找到你多次使用的字体


Blog Account
SoftwareTeacher@cnbl...
Verdana
9.8
Publish
Paragraph
Agency FB
Aharoni
Algerian
Andalus
Andy
Angsana New
AngsanaUPC
Aparajita
Arabic Typesetting
Arial
Arial Black
Arial Narrow
Arial Rounded MT Bold
Arial Unicode MS
Baskerville Old Face
Batang
BatangChe
Bauhaus 93
Bell MT
Berlin Sans FB
Berlin Sans FB Demi
Bernard MT Condensed
Blackadder ITC
Bodoni MT
Bodoni MT Black
Bodoni MT Condensed
Bodoni MT Poster Compressed
Book Antiqua
a) 用得越多，一样难用，
b) 用得越多，更加难用，
c) 用得越多，越来越好用？

Word 字体选择

- Microsoft Office Word 就有一个更好的设计，它把字体划分为三个档次，由上而下地显示出来：
  - 当前Word模板的主题字体；
  - 最近使用的字体；
  - 所有字体。



短期/长期

- 如果你要在电脑前工作两分钟，你希望用什么控制电脑
  - 鼠标键盘。
  - 用手指在屏幕上操作。
  - 带上专用手套，启动摄像头，用手势操作。
  - 语音。
- 如果你要在电脑前工作30 分钟呢？
- 8小时呢？

5W1H

- **Who：** 谁是你的目标用户？
- **When：** 他们会在什么时间使用你的产品？比如一个邮件应用，用户在起床时可能更偏向于快 速查看，而在工作时间会发生更多的输入操作。
- **Where：** 目标用户会在哪里和你的产品交互？是晃动的公交车上或阳光耀眼的室外？还是在沙 发上？
- **What：** 你的产品是什么？而用户的期待是什么？
- **Why：** 用户为什么要使用你的产品？他们的动机是什么？还有，在众多竞争产品中，用户为 什么会选择你的产品？
- **How：** 用户是如何与你的产品发生交互的？他们怎么用？在使用过程中出现了什么问题吗？

设计的三个层次

设计的三个层次，以及对应的产品特性：

- 本能（Visceral）层次的设计—外形
- 行为（Behavior）层次的设计—使用的乐趣和效率
- 反思（Reflective）层次的设计—自我形象、个人满足感、回忆

你家里挂有蒙娜丽莎的油画：

- 100% 完美的电子版，在平板显示器上
- 100% 完美的复制品，在画布上
- 原作

这三个选择给你家的访客什么不同的感受？

原则

- **1．尽快提供可感触的反馈**
- 系统状态要有反馈，等待时间要合适。现在程序发生了什么，应该在某一个统一的地方清晰地标示出来。一个目标用户能够只靠软件的主要反馈来完成基本的操作，而不用事先学习使用手册。系统的反馈可以是视觉的、听觉的、触觉的（例如手机发生的振动）。

- **2．系统界面符合用户的现实惯例（Familiarity，Avoid surprise）**
- 软件系统要用用户语言，而不是开发者语言来和用户沟通，所用的概念要贴近实际生活，而不是用学术概念或开发者的概念。我们说的生活实际，最好是目标用户的实际生活体验。例如，给病人使用的网络挂号系统，就不宜使用只有医务工作者才熟悉的术语和界面。（最坏的结果是使用软件工程师才熟悉的术语和界面，而医护人员和病人对此很不熟悉）。软件的反馈要避免带给用户惊奇——例如，在用户没有期待对话框的时候，软件从奇怪的角度弹出对话框。或者给用户提示“找不到对象”。

- **3. 用户有自由控制权**
- 操作失误可回退，要让用户可以退出软件（很多软件都没有退出菜单，这是导致用户反感的一大来源）。用户可以定制显示信息的多少，还可以定制常用的设置。

- **4. 一致性和标准化**
- 软件中对同一事物和同类操作的表示用语，各处要保持一致。例如，某词典软件有“帮助用户收集生词并且背诵生词”的功能。这个功能要有明确一致的称呼，不能混杂着叫“单词本”、“生词本”、“Word List”、“Word Book”、“单词文件”……等等。

- **5. 适合各种类型的用户**
- 我们的软件要为新手和专家提供可定制化的设计。一些操作方式，快捷操作可调整。我们还应该为某些障碍的用户（色弱、色盲、盲人、听力有缺陷的用户、操作键盘鼠标不方便的用户等等）提供一定程度的便利。对于长期使用一个软件的用户，软件应该能适应用户的使用习惯，让用户越用越顺手。

- **6．帮助用户识别、诊断并修复错误**
- 软件的关键操作需要有确认提示，以便帮助用户及早消除误操作。要注意使用朴素的语言来表述错误信息。错误信息需要给出下一步操作提示（我现在出错了，那下一步怎么办？）。必要时提供详细的帮助信息，并协助用户方便地从错误中恢复工作。
- 让所有的用户都可以通过电子邮件或者提交表单来提交反馈意见。有些程序用一对简单的笑脸/哭脸符号来鼓励用户提交反馈，这也是很好的办法。

需要文档么？

- 有必要的提示和帮助文档
- 无需文档就能流畅应用当然更好，如果必要的话，可提供一个在线帮助。如果软件和用户的工作相关（而不是简单的游戏），那么基本的提示和帮助文档还是很有必要的，而且也要提供便利的检索功能。文档要从用户的角度出发描述具体步骤，并且不要太冗长。

讨论：

产品设计的细节:网页、PC软件和手 机软件有许多地方都会出现下面的两个按钮:

确定 | 取消

Ok | Cancel

- 这两个小小的按钮也大有文章：[确定] 按钮是放 在左边还是右边？哪一个按钮是处于预先选择的状态（按回车键的时候就自动选择）？哪一种设计更符合人类习惯？你觉得这个问题重要么？你怎么设计统一的规范？

- 你觉得是用OK/Cancel的按钮选择好呢？还是在按钮上标明动作如[退出]/ [保存] ？