



大连海事大学
DALIAN MARITIME UNIVERSITY

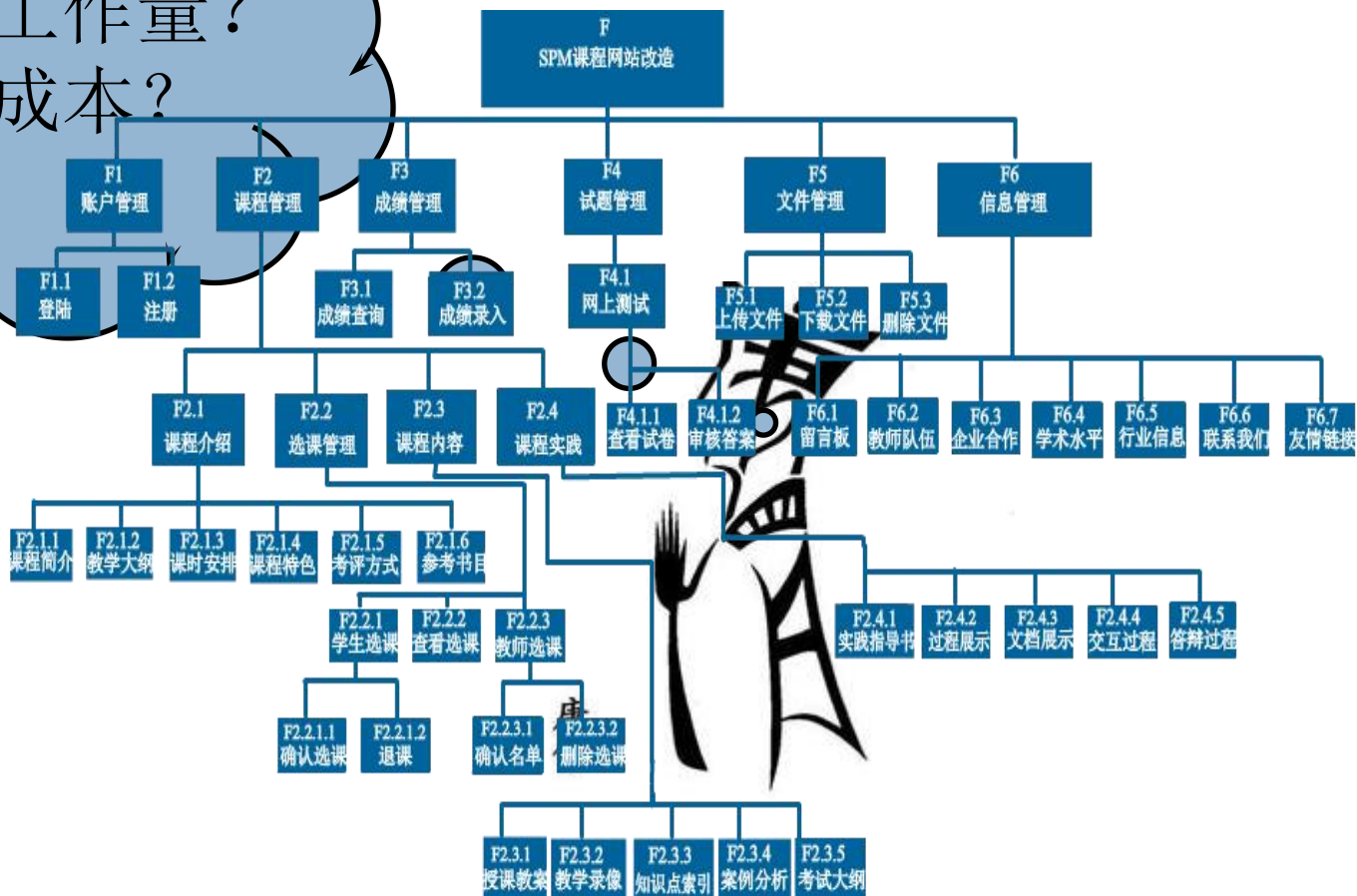
工程（软件）项目管理

信息科学技术学院 伍延斌
E-mail: top32@163.com

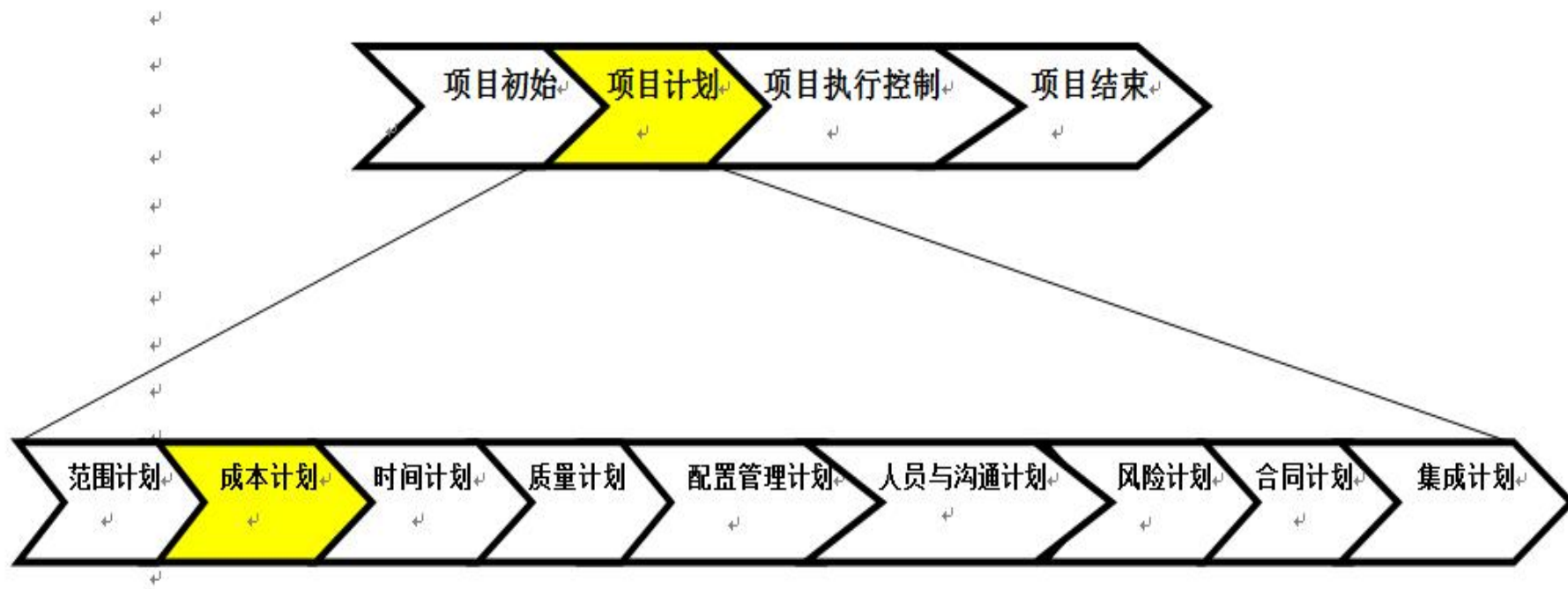
Information Science and Technology College
Dalian Maritime University

情景引入

多少工作量？
多少成本？



成本规划



软件项目管理



第二篇

第 6 章

软件项目成本计划

本章要点

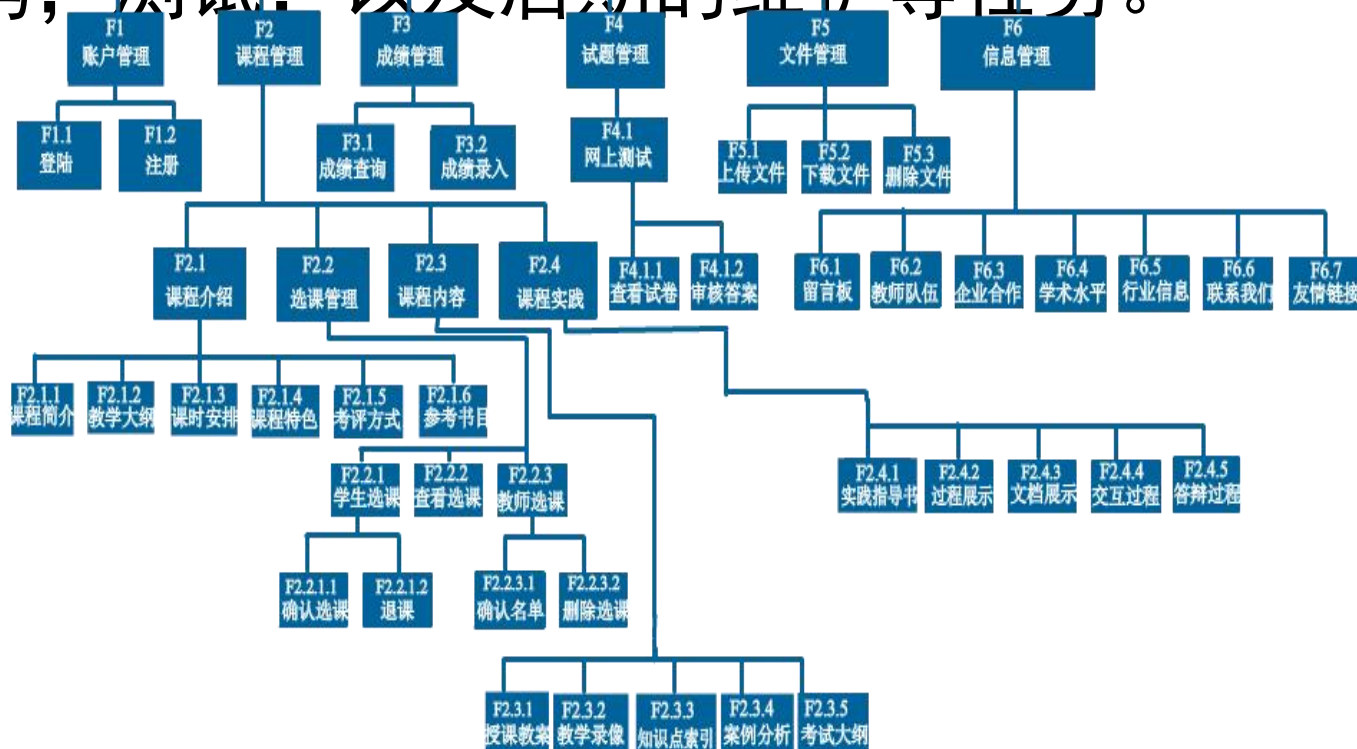
- ❑ 一、估算过程概念
- ❑ 二、估算方法
- ❑ 三、成本预算
- ❑ 四、案例分析
- ❑ 五、课程实践

6.1.1 关于估算

- ❑ 估算不是很准确，有误差
- ❑ 项目经验数据非常重要
- ❑ 不要太迷信某些数学模型

6.1.2-1 软件项目规模

- ❑ 软件项目规模即工作量
- ❑ 例如：软件规划，**SPM课程网站改造**管理，需求，设计，编码，测试，以及后期的维护等任务。



6.1.2-2 软件规模单位

- ❑ LOC (Loc of Code)
 - ❑ 源代码长度的测量
- ❑ FP (Function Point)
 - ❑ 用系统的功能数量来测量
- ❑ 人月
- ❑ 人天
- ❑ 人年

6.1.3-1 软件项目成本

- ❑ 完成软件规模相应付出的代价。
- ❑ 待开发的软件项目需要的资金。
- ❑ 人的劳动的消耗所需要的代价是软件产品的主要成本



6. 1. 3-2 成本的单位

- ❑ 货币单位
 - ❑ 人民币元
 - ❑ 美元
 - ❑



6.1.3-3 软件规模和软件成本的关系

- ❑ 规模是成本的主要因素，是成本估算的基础
- ❑ 有了规模就确定了成本

6. 1. 4-1 成本估算结果

- ❑ 直接成本
- ❑ 间接成本

6.1.4-2 直接成本

- ❑ 与具体项目相关的成本,
- ❑ 例如：参与项目的人员成本

直接成本 = 每个开发者成本的合计

每个开发者成本 =

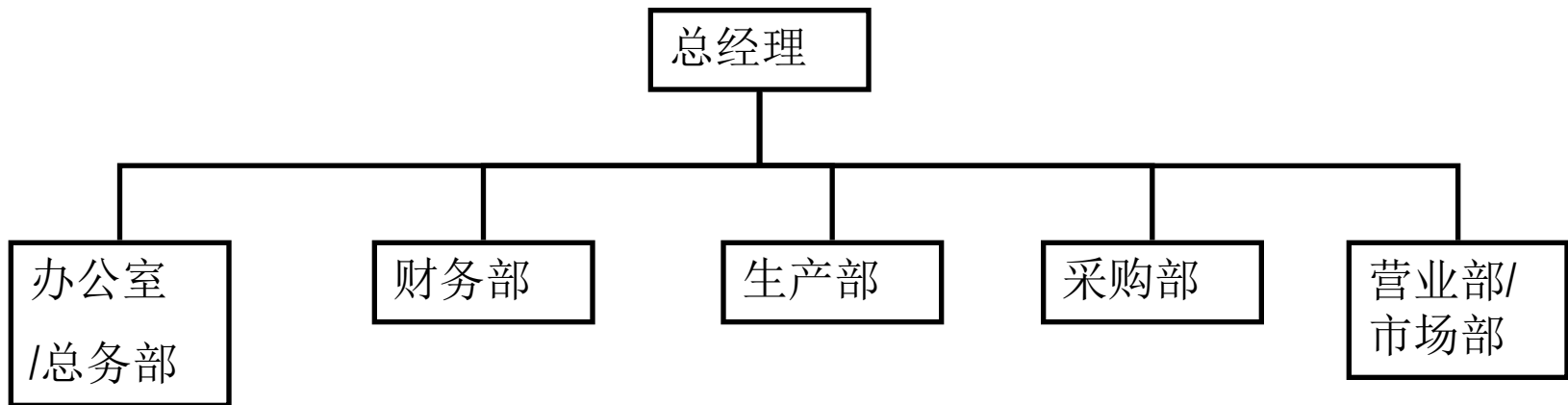
开发时间（规模） ×
人工单价

6.1.4-3 间接成本

- ❑ 可以分摊到各个具体项目中的成本，例如：
 - ❑ 培训
 - ❑ 房租水电
 - ❑ 员工福利
 - ❑ 市场费用
 - ❑ 管理费
 - ❑ 其他等等

直接成本（补充）

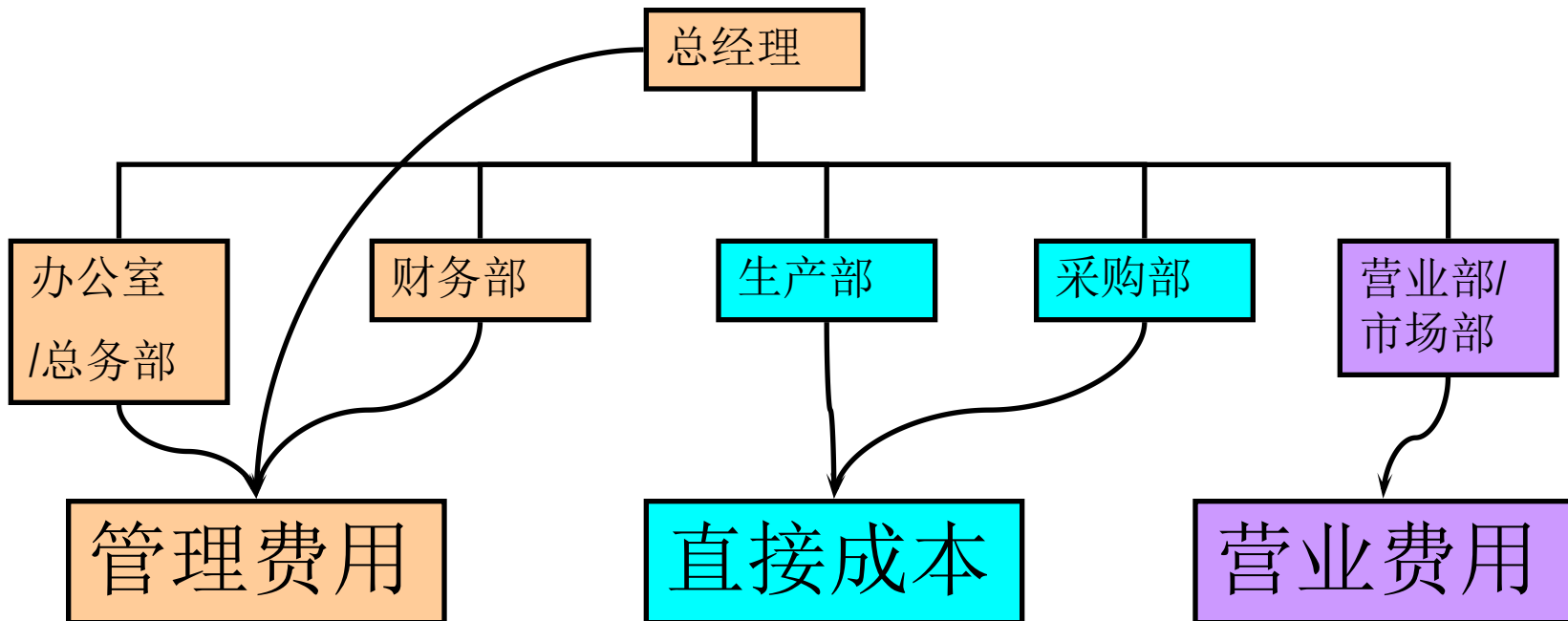
- ❑ 不是所有人员的工资支出都是直接成本



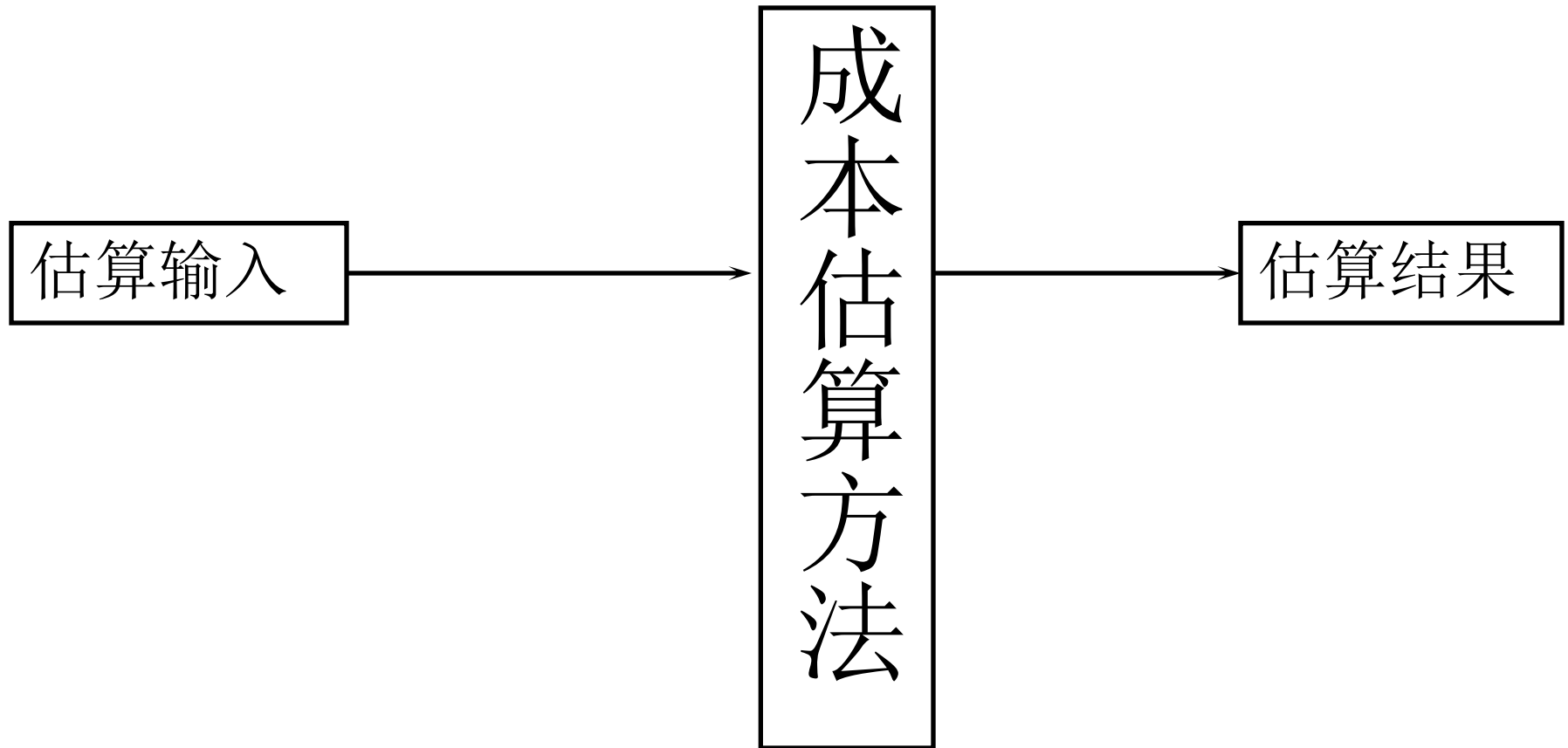
- ❑ 哪些部门的工资属于直接生产成本？
- ❑ 不属于直接生产成本的部门工资是什么费用？

直接成本（补充）

- ❑ 不是所有人员的工资支出都是直接成本



6.1.5-1 成本估算过程



6.1.5-2 成本估算输入

- ❑ 项目需求、 WBS
- ❑ 历史项目度量
- ❑ 资源要求（资源编制计划）
- ❑ 资源消耗率：如人员成本：100元/小时
- ❑ 进度规划：项目总进度（一般是合同要求）
- ❑ 学习曲线

6.1.5-3 资源规划

□ 需要的资源种类、数量等

比如开发团队的构成

- 社内员工（PM/PL/BSE/SE/PG, TE, QC）
- 外协人员

课内讨论：软件公司为什么要使用外协人员？

6.1.5-4 项目估算结果

- ❑ 估算文件
 - ❑ 资源，资源的数量，质量标准，估算成本等信息
 - ❑ 单位：一般是货币单位
 - ❑ BAC (Budget At completion)
- ❑ 估算说明
 - ❑ 工作范围
 - ❑ 估算的基础和依据
 - ❑ 估算的假设
 - ❑ 估算的误差变动等

6.1.5-5 估算说明

- ❑ 预测所需要的总工作量的过程。
- ❑ 是一种量化的结果
- ❑ 可以有一些误差
- ❑ 成本估算不同于项目定价
- ❑ 贯穿于软件的生存周期。

本章要点

- ❑ 一、估算过程概念
- ❑ 二、估算方法
- ❑ 三、成本预算
- ❑ 四、案例分析
- ❑ 五、课程实践

6.2 估算基本方法



1. 代码行估算法
2. 功能点估算法
3. 用例点估算法
4. 类比（自顶向下）估算法
5. 自下而上估算法
6. 参数估算法
7. 专家估算法

6.2.1-1 代码行估算法

从软件程序量的角度定义项目规模。

- ❑ 与具体的编程语言有关
- ❑ 分解足够详细
- ❑ 有一定的经验数据（类比和经验方法）

6.2.1-2 代码行技术的主要优点

- 1) 代码是所有软件开发项目都有的“产品”，而且很容易计算代码行数。
- 2) 有大量的生产性数据为依据，因此基于代码行的估算精度比较准确

6.2.1-3 代码行估算的缺点

1. 对代码行没有公认的可接受的标准定义
2. 代码行数量依赖于所用的编程语言和个人的编程风格.
3. 在项目早期,需求不稳定、设计不成熟、实现不确定的情况下很难准确地估算代码量.
4. 代码行强调编码的工作量,只是项目实施阶段的一部分

升级改造项目比较适合用
代码行估算法

代码行补充

问题：

一个程序员究竟每天能写多少行代码？

代码行补充

问题：

一个程序员究竟每天能写多少行代码？

- 一般在300行-500行（.NET和Java程序员）
- 根据语言种类及开发要求不同有所不同
- 比喻有些程序要求注释较多，则代码就多
- SAP ABAP程序员每个月的代码行在4000行左右，
其开发工作还包括测试和测试文档作成。

估算的基本方法

1. 代码行估算法



功能点估算法

3. 用例点估算法

4. 类比（自顶向下）估算法

5. 自下而上估算法

6. 参数估算法

7. 专家估算法

6.2.2-1 功能点估算

- ❑ 与实现的语言和技术没有关系
- ❑ 用系统的功能数量来测量其规模
- ❑ 通过评估、加权、量化得出功能点

6.2.2-2 功能点公式

$$FP = UFC * TCF$$

- UFC：未调整功能点计数
- TCF：技术复杂度因子

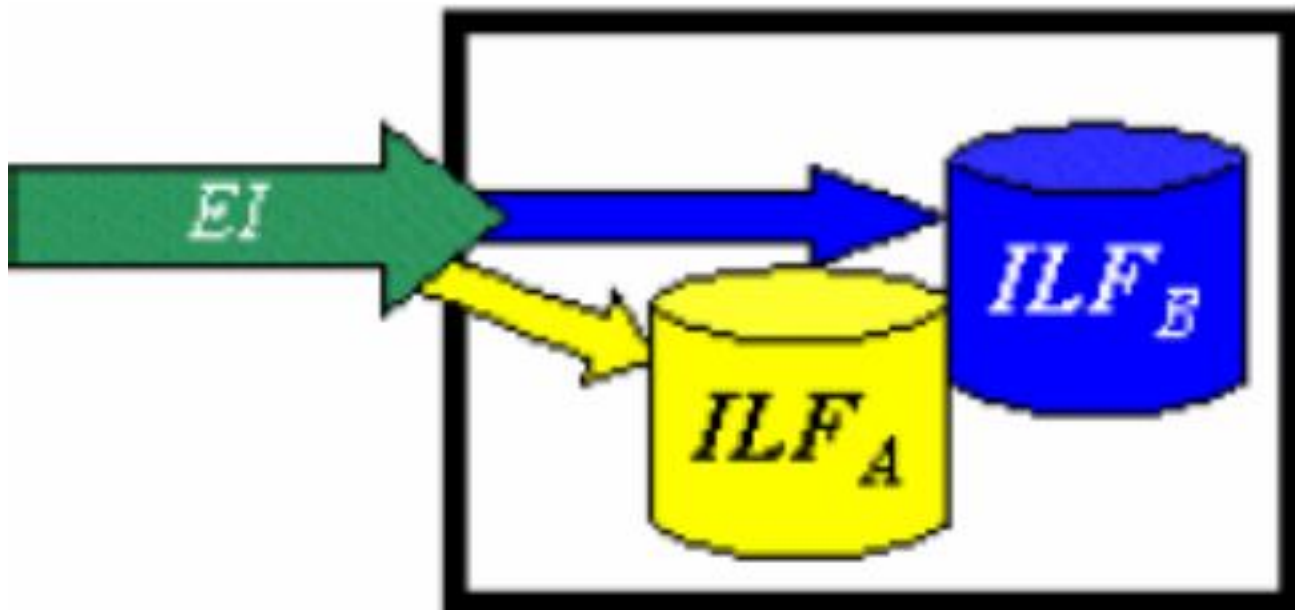
6. 2. 2-3 UFC-未调整功能点计数

功能计数项：(从处理逻辑的角度)

1. 外部输入
2. 外部输出
3. 外部查询
4. 外部接口文件
5. 内部逻辑文件

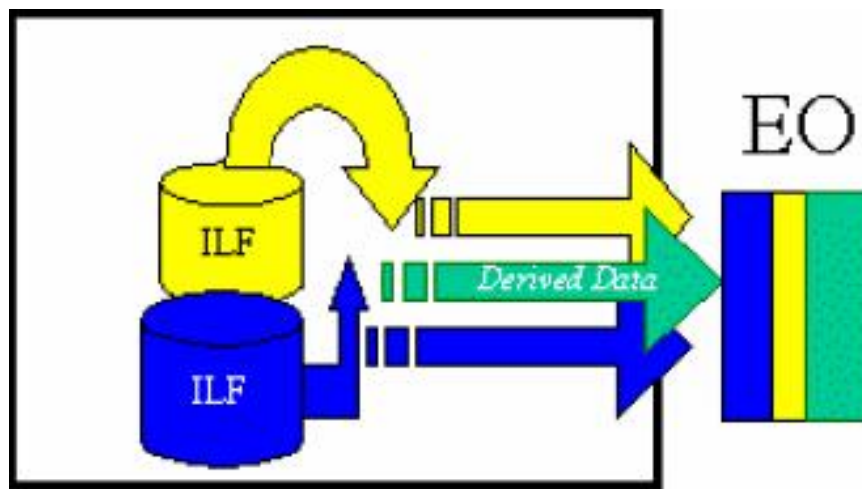
外部输入 (External Inputs: EI)

给软件提供面向应用的数据的项（如屏幕、表单、对话框、控件，文件等）；在这个过程中，数据穿越外部边界进入到系统内部。



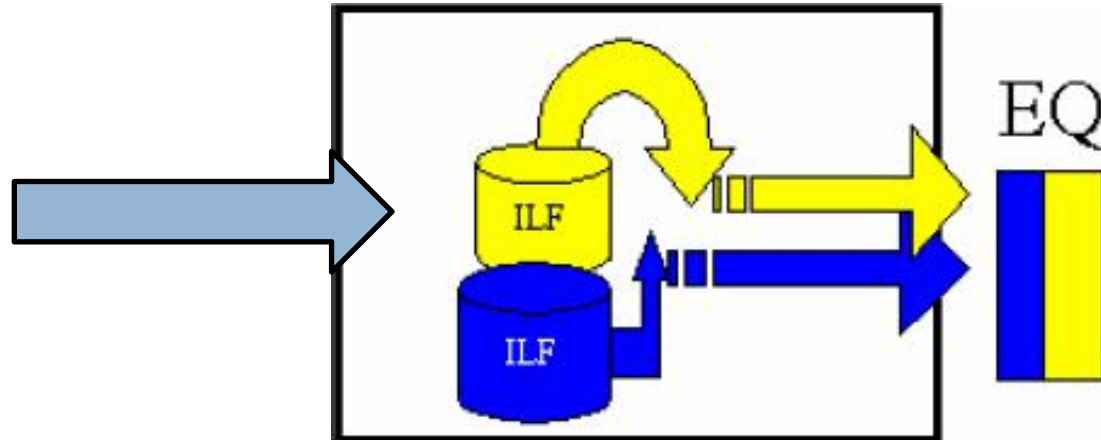
外部输出(External Outputs EO)

向用户提供(经过处理的)面向应用的信息，例如，报表和出错信息等。



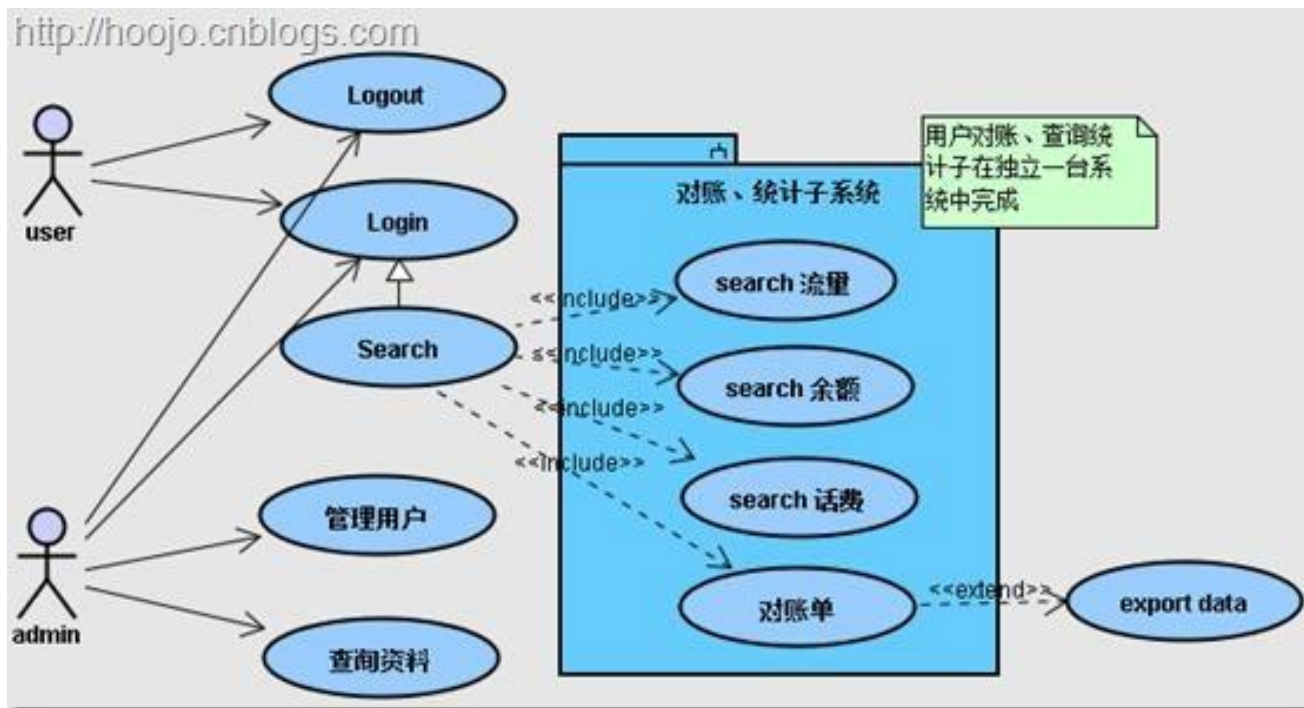
外部查询(External Inquiry EQ)

外部查询是一个输入引出一个即时的简单输出。
没有处理过程。



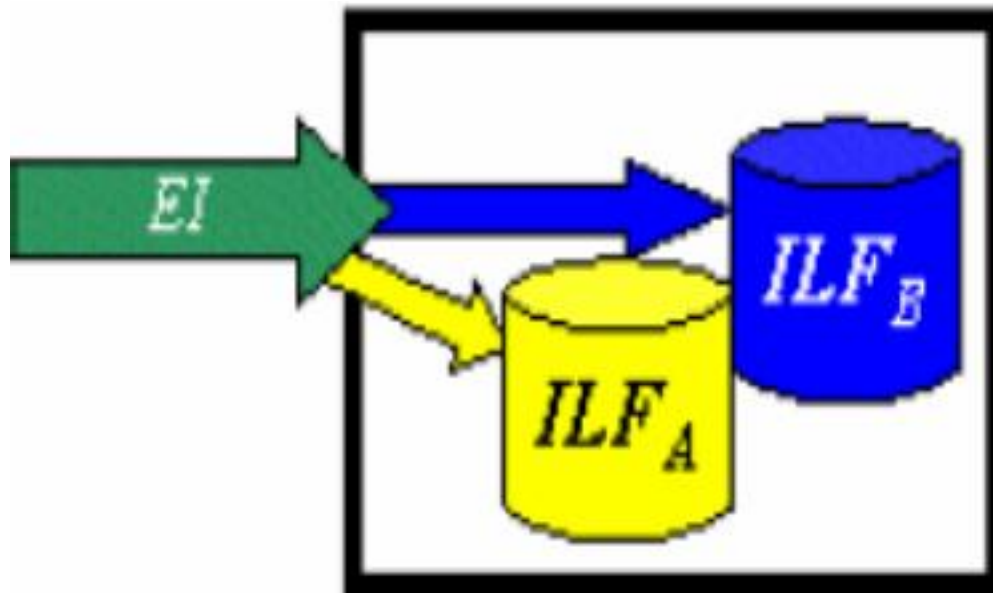
外部接口文件 (External Interface Files EIF's)

外部接口文件是用户可以识别的一组逻辑相关数据，这组数据只能被引用。用这些接口把信息传送给另一个系统。



内部逻辑文件(Internal Logical Files: ILF'S)

用户可以识别的一组逻辑相关的数据，而且完全存在于应用的边界之内，并且通过外部输入维护，是逻辑主文件的数目。

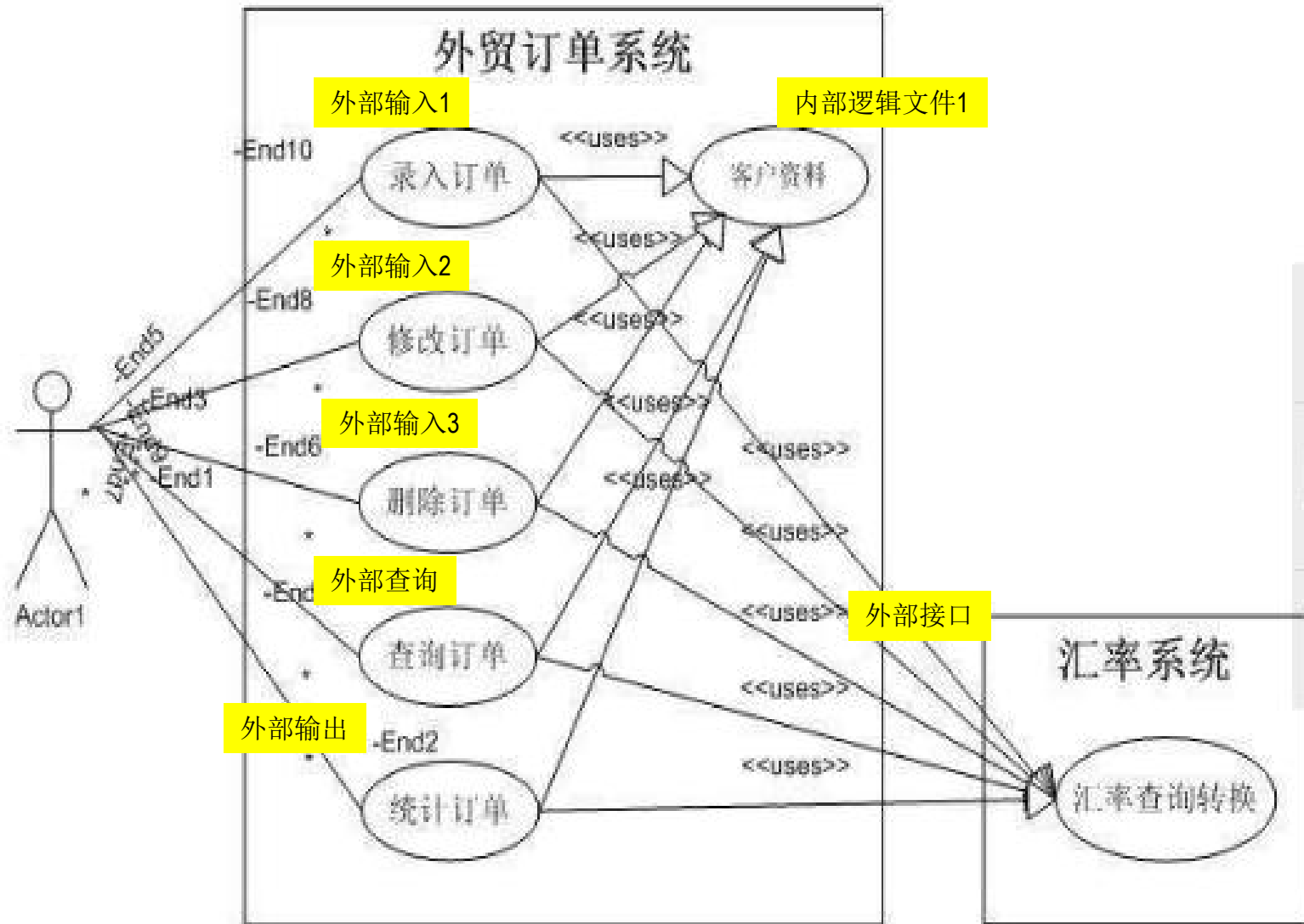


6.2.2-4 UFC-未调整功能点计数

功能计数项的复杂度等级

项	复杂度权重因素		
	简单(低)	一般（中）	复杂（高）
外部输入	3	4	6
外部输出	4	5	7
外部查询	3	4	6
外部接口文件	5	7	10
内部逻辑文件	7	10	15

6.2.2-5 FP估算方法举例



功能点计算实例-UFC

根据上面的外贸订单项目的需求评估:

外部输入:3项; 外部输出:1项; 外部查询:1项;

外部接口文件:1项; 内部逻辑文件:2项 (客户和订单)

	功能点		
项	简单	一般	复杂
外部输入	2 * 3	1 * 4	0 * 6
外部输出	0 * 4	0 * 5	1 * 7
外部查询	0 * 3	1 * 4	0 * 6
外部接口文件	0 * 5	1 * 7	0 * 10
内部逻辑文件	1 * 7	1 * 10	0 * 15
总计			
UFC	45		

TCF-技术复杂度因子（14个）

$TCF = 0.65 + 0.01(\sum(F_i))$: $F_i: 0-5$, $TCF: 0.65-1.35$

技术复杂度因子			
F1	可靠的备份和恢复	F2	数据通信
F3	分布式函数	F4	性能
F5	大量使用的配置	F6	联机数据输入
F7	操作简单性	F8	在线升级
F9	复杂界面	F10	复杂数据处理
F11	重复使用性	F12	安装简易性
F13	多重站点	F14	易于修改

技术复杂度因子的取值范围

调整系数	描述
0	不存在或者没有影响
1	不显著的影响
2	相当的影响
3	平均的影响
4	显著的影响
5	强大的影响

外贸订单项目：功能点计算实例

- $FP = UFC * TCF$

- $UFC = 45$

- $TCF = 0.65 + 0.01(14 * 3) = 1.07$ （平均技术复杂因子）

- $FP = 45 * 1.07 = 48$

- 如果： $PE = 15$ 工时/功能点 （生产性）


- 则： $Effort = 48 * 15 = 720$ 工时 （规模）

（ $720H = 90$ 人天 = 4.5 人月）

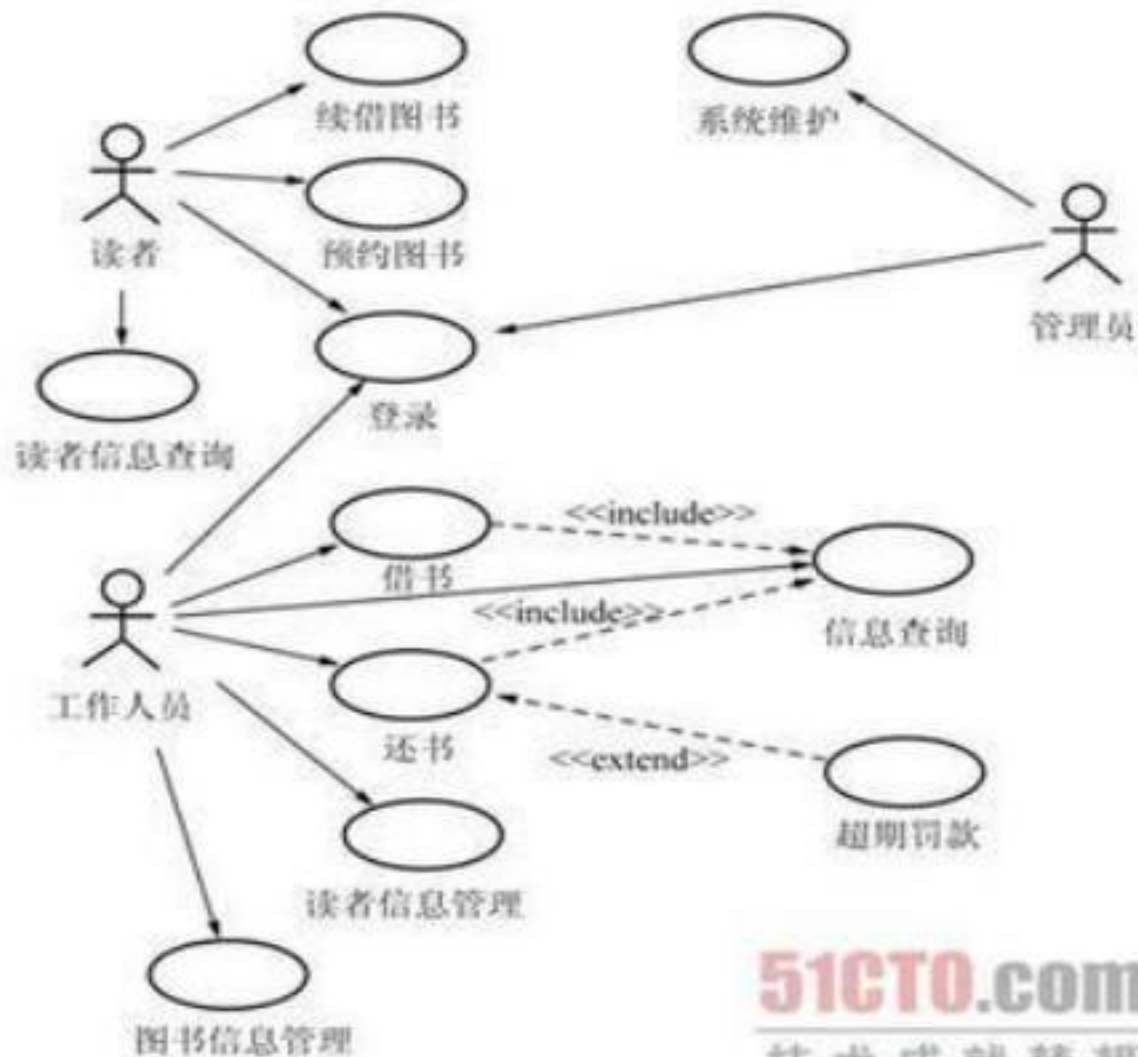
功能点与代码行的转换

语言	代码行/FP
Assembly	320
C	150
COBOL	105
FORTRAN	105
PASCAL	91
ADA	71
PL/1	65
PROLOG/LISP	64
SMALLTALK	21
SPREADSHEET	6

估算的基本方法

1. 代码行估算法
2. 功能点估算法
-  用例点估算法
4. 类比（自顶向下）估算法
5. 自下而上估算法
6. 参数估算法
7. 专家估算法

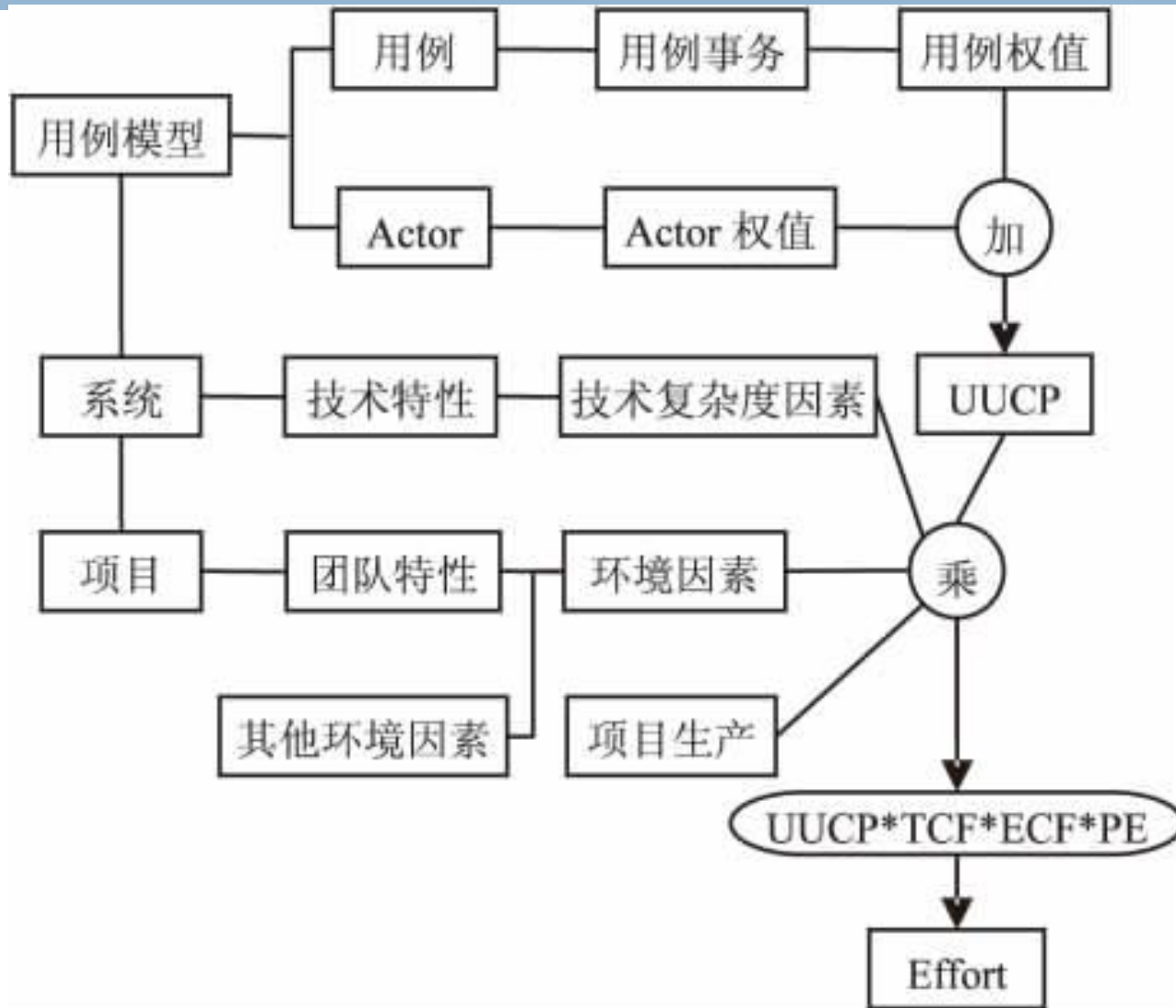
6.2.3-1 用例模型



角色

用例

6.2.3-2 用例点估算模型



用6.2.3-3例点估算方法的基本步骤

1. 计算未调整的角色权值UAW;
2. 计算未调整的用例权值UUCW ;
3. 计算未调整的用例点UUCP;
4. 计算技术和环境因子TEF;
5. 计算调整的用例点UCP ;
6. 计算工作量(man—hours) 。

1、计算未调整的角色权值UAW

$$UAW = \sum_{c \in C} aWeight(c) \times aCardinality(c)$$

$aWeight(c)$: 权重的权值

$aCardinality(c)$: 基数, 参与的角色数目

表6-12 Actor 权值定义

序号	复杂度级别	复杂度标准	权值
1	simple	角色通过API 与系统交互	1
2	average	角色通过协议与系统交互	2
3	complex	用户通过GUI 与系统交互	3

2、计算未调整的用例的权值UUCW

$$UUCW = \sum_{c \in C} uWeight(c) \times uCardinality(c)$$

表6-13 Use Case 权值定义

序号	复杂度级别	事务/场景个数	权值
1	simple	1—3	5
2	average	4—7	10
3	complex	>7	15

3、计算未调整的用例点UUCP

$UUCP = UAW + UUCW$: 例如

表6-16: Actor 权值

序号	复杂度级别	权值	参与角色数	UAWi
1	simple	1	2	2
2	average	2	4	8
3	complex	3	5	15

25

表6-17: Use Case 权值

序号	复杂度级别	权值	用例数	UUCWi
1	simple	5	5	25
2	average	10	2	20
3	complex	15	3	45

85

$$UAW = 1 \times 2 + 2 \times 4 + 3 \times 5 = 2 + 8 + 15 = 25$$

$$UUCW = 5 \times 5 + 10 \times 2 + 15 \times 3 = 25 + 20 + 45 = 85$$

$$UUCP = UAW + UUCW = 25 + 85 = 110$$

4、计算技术因子TCF

$$TCF = 0.6 + (0.01 \times \sum_{i=1}^{13} TCF_Weight_i \times Value_i)$$

表6-14 技术复杂度因子的定义

序号	技术因子	说明	权值
1	TCF1	分布式系统	2.0
2	TCF2	性能要求	1.0
3	TCF3	最终用户使用效率	1.0
4	TCF4	内部处理复杂度	1.0
5	TCF5	复用程度	1.0
6	TCF6	易于安装	0.5
7	TCF7	系统易于使用	0.5
8	TCF8	可移植性	2.0
9	TCF9	系统易于修改	1.0
10	TCF10	并发性	1.0
11	TCF11	安全功能特性	1.0
12	TCF12	为第三方系统提供直接系统访问	1.0
13	TCF13	特殊的用户培训设施	1.0

$$TCF = 0.6 + 0.01 \times (2.0 \times 3 + 1.0 \times 5 + 1.0 \times 3 + 1.0 \times 5 + 1.0 \times 0 + 0.5 \times 3 + 0.5 \times 5 + 2.0 \times 3 + 1.0 \times 5 + 1.0 \times 3 + 1.0 \times 5 + 1.0 \times 0 + 1.0 \times 0) = 1.02$$

4、 计算环境因子ECF

$$ECF = 1.4 + (-0.03 \times \sum_{i=1}^8 ECF_Weight_i \times Value_i)$$

表6-15 环境因子的定义

序号	环境因子	说明	权值
1	ECF1	UML 精通程度	1.5
2	ECF2	系统应用经验	0.5
3	ECF3	面向对象经验	1.0
4	ECF4	系统分析员能力	0.5
5	ECF5	团队士气	1.0
6	ECF6	需求稳定度	2.0
7	ECF7	兼职人员比例高低	1.0
8	ECF8	编程语言难易程度	1.0

$$\begin{aligned} ECF &= 1.4 + (-0.03 \times (1.5 \times 3 + 0.5 \times 3 + 1.0 \times 3 + 0.5 \times 5 + 1.0 \times 3 + 2.0 \times 3 + 1.0 \times 0 + 1.0 \times 0)) \\ &= 0.785 \end{aligned}$$

5、计算调整的用例点UCP

$$UCP = UUCP \times TCF \times ECF$$

$$\begin{aligned} UCP &= UUCP \times TCF \times ECF \\ &= 110 \times 1.02 \times 0.785 = 88 \end{aligned}$$


6、计算工作量

$$Effort = UCP \times PF$$

如果：**PF = 20**工时/用例点

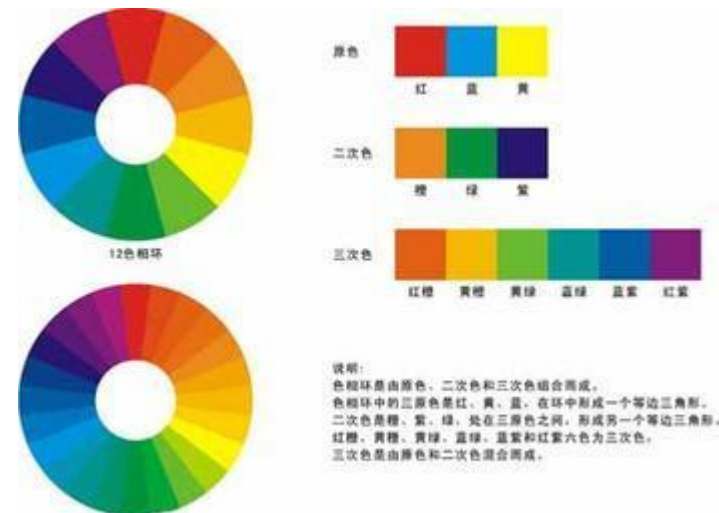
则：**Effort = UCP × PF = 88 × 20 = 1760h = 220**人天

估算的基本方法

1. 代码行估算法
2. 功能点估算法
3. 用例点估算法
-  4. 类比（自顶向下）估算法
5. 自下而上估算法
6. 参数估算法
7. 专家估算法

6.2.4-1 类比估算-定义

- ❑ 估算人员根据以往的完成类似项目所消耗的总成本（或工作量），来推算将要开发的软件的总成本（或工作量），然后按比例将它分配到各个开发任务单元中
- ❑ 是一种自上而下的估算形式



6. 2. 4-2 类比估算—使用情况

- ❑ 有类似的历史项目数据
- ❑ 信息不足（例如市场招标）的时候
- ❑ 要求不是非常精确估算的时候

6.2.4-3 类比估算—理论举例

例 3: 一个待估算工作量的项目 P_0 和已完成的项目 P_1, P_2 .

Project	Project type	Programming language	Team size	Project size	Effort
P_0	MIS	C	9	180	
P_1	MIS	C	11	200	1 000
P_2	Real time	C	10	175	900


项目间的相似度计算如下:

P_0 vs. P_1	P_0 vs. P_2
$\delta(P_{01}, P_{11}) = \delta(\text{MIS}, \text{MIS}) = 0$	$\delta(P_{01}, P_{21}) = \delta(\text{MIS}, \text{Real Time}) = 1$
$\delta(P_{02}, P_{12}) = \delta(\text{C}, \text{C}) = 0$	$\delta(P_{02}, P_{22}) = \delta(\text{C}, \text{C}) = 0$
$\delta(P_{03}, P_{13}) = \delta(9, 11)$ $= [(9-11)/(9-11)]^2 = 1$	$\delta(P_{03}, P_{23}) = \delta(9, 10)$ $= [(9-10)/(9-11)]^2 = 0.25$
$\delta(P_{04}, P_{14}) = \delta(180, 200)$ $= [(180-200)/(200-175)]^2 = 0.64$	$\delta(P_{04}, P_{24}) = \delta(180, 175)$ $= [(180-175)/(200-175)]^2 = 0.04$
$\text{distance}(P_0, P_1) = (1.64/4)^{0.5} = 0.64$	$\text{distance}(P_0, P_2) = 0.57$

类比估算—主观判断举例

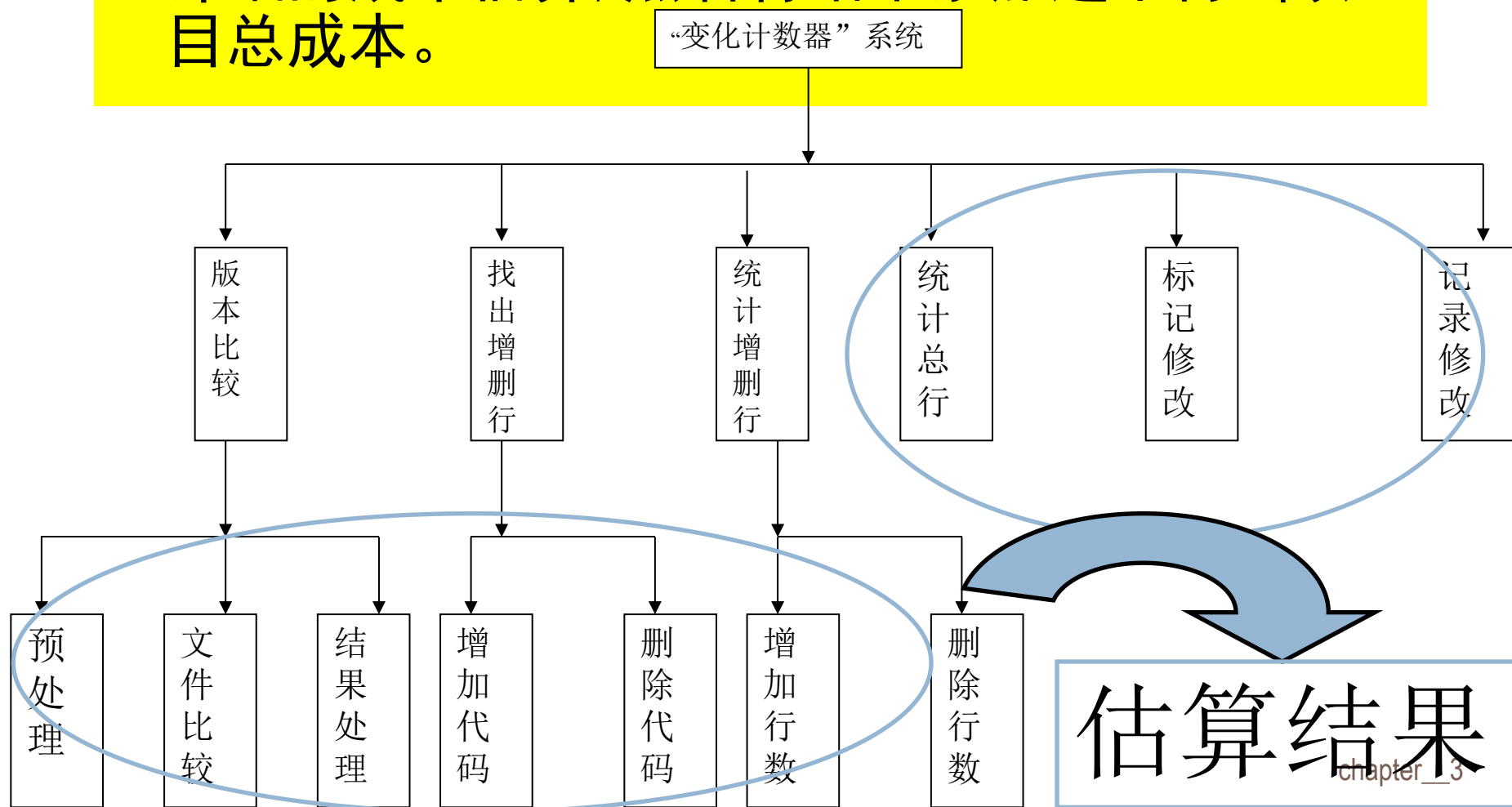
- ❑ 证券交易网站
 - ❑ 需求类似
 - ❑ 历史数据：10万
 - ❑ 类比估算：10万

估算的基本方法

1. 代码行估算法
2. 功能点估算法
3. 用例点估算法
4. 类比（自顶向下）估算法
-  5. 自下而上估算法
6. 参数估算法
7. 专家估算法

6.2.5-1 自下而上估算——定义

- 利用任务分解图(WBS), 对各个具体工作包进行详细的成本估算, 然后将结果累加起来得出项目总成本。



6.2.5-2 自下而上估算——特点

- ❑ 相对比较准确，它的准确度来源于每个任务的估算情况
- ❑ 花费时间

自下而上估算举例


M（管理）、D（交付）、
Q（质量）、S（支持或安全）

综合业务系统成本估算表如下：

+

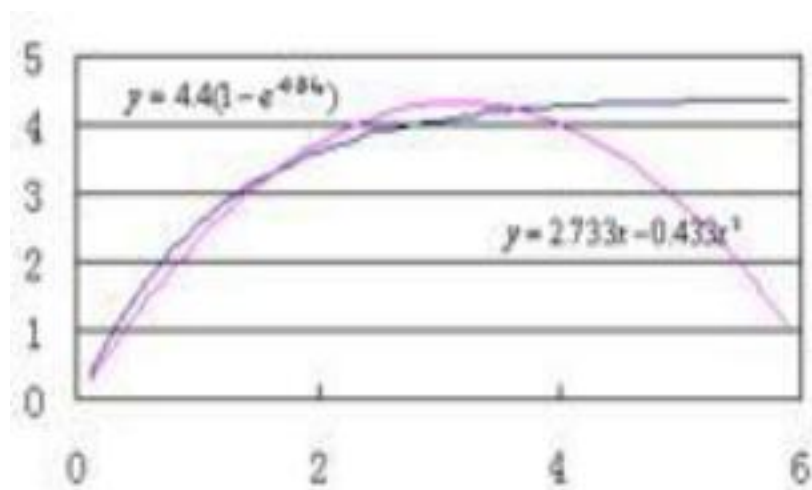
阶段	人力	时间 (月)	成本 (万)	总计 (万)
项目准备阶段	M:2/D:8/Q:1	0.5	16.5	145.5
设计阶段	M:2/D:8/Q:2/ S:1	1	39	
基础模块开发：				
公共控制子系统		1.5	90	
中央会计子系统	M:2/D:15/Q:2/ S:1			126
客户信息子系统				
基本功能模块开发：				
帐户管理子系统	M:2/D:12/Q:2/ S:1	0.25	12.75	
出纳管理子系统	M:2/D:18/Q:2/ S:1	0.5	34.5	189.75
凭证管理子系统	M:2/D:8/Q:2/ S:1	0.25	9.75	
会计核算子系统	M:2/D:18/Q:2/ S:1	0.5	34.5	
储蓄子系统	M:2/D:18/Q:2/ S:1	0.5	34.5	
扩展功能模块开发：				42
同城业务子系统	M:2/D:15/Q:2/ S:1	0.25	15	
联行业务子系统	M:2/D:18/Q:2/ S:1	0.5	34.5	
内部清算子系统	M:2/D:12/Q:2/ S:1	0.25	12.75	
固定资产管理子系统	M:2/D:10/Q:2/ S:1	0.25	11.25	
信贷管理子系统	M:2/D:18/Q:2/ S:1	0.5	34.5	
一卡通业务子系统	M:2/D:18/Q:2/ S:1	0.5	34.5	
中间业务子系统	M:2/D:12/Q:2/ S:1	0.25	12.75	
金卡接口模块	M:2/D:18/Q:2/ S:1	0.5	34.5	
现场联调	M:2/D:10/Q:2	1	42	42
	503.25			

估算的基本方法

1. 代码行估算法
2. 功能点估算法
3. 用例点估算法
4. 类比（自顶向下）估算法
5. 自下而上估算法
-  6. 参数估算法
7. 专家估算法

6.2.6-1 参数估算法—定义

- ❑ 通过项目数据, 进行回归分析, 得出回归模型
(根据大量的实际项目的数据建立模型。)
- ❑ 通过参数模型估算(规模)成本的方法。



参数6.2.6-2 估算法—使用条件

- ❑ 具有良好的项目数据为基础
- ❑ 存在成熟的项目估算模型

6.2.6-3 参数估算法一特点

- ❑ 比较简单, 而且也比较准确
- ❑ 如果模型选择不当或者数据不准, 也会导致偏差

6.2.6-4参数模型: 规模(成本)模型

面向LOC(代码行)驱动的

- ❑ Walston-Felix (IBM)
 - ❑ $E = 5.2 * (KLOC)^{0.91}$
- ❑ Bailey-Basili
 - ❑ $E = 5.5 + 0.73 * (KLOC)^{1.16}$
- ❑ .COCOMO
 - ❑ $E = 3.2 * (KLOC)^{1.05}$
- ❑ Doty
 - ❑ $E = 5.288 * (KLOC)^{1.047}$

参数模型：规模（成本）模型

面向FP（功能点）驱动的

- ❑ Albrecht and Gaffney

- ❑ $E = -12.39 + 0.0545FP$


- ❑ Matson, Barnett

- ❑ $E = 585.7 + 15.12FP$

参数模型：规模（成本）模型

- ❑ 整体公式： $E=a+b*S^c$
 - ❑ E：以人月表示的工作量
 - ❑ a, b, c：经验导出的系数
 - ❑ S：主要的输入参数（通常是LOC, FP等）

估算的基本方法

1. 代码行估算法
 2. 功能点估算法
 3. 用例点估算法
 4. 类比（自顶向下）估算法
 5. 自下而上估算法
 6. 参数估算法
-  专家估算法

6.2.7-1 专家估算法

- 由多位专家进行成本估算，一个专家可能会有偏见，最好由多位专家进行估算，取得多个估算值，最后得出综合的估算值。

6.2.7-2 专家估算法-Deiphi

- ❑ 组织者确定专家，这些专家互相不见面
- ❑ 组织者发给每位专家一份软件规格说明
- ❑ 专家以无记名对该软件给出3个规模的估算值
 - ❑ 最小 a_i
 - ❑ 最可能的 m_i
 - ❑ 最大 b_i
- ❑ 组织者计算每位专家的 $E_i = (a_i + 4m_i + b_i) / 6$
- ❑ 如果各个专家的估算差异超出规定的范围（例如：15%），则需重复上述过程
- ❑ 最终可以获得一个多数专家共识的软件规模： $E = (E_1 + E_2 + \dots + E_n) / n$ （N:表示N 个专家）

6.2.7-3 Deiphi 专家估算法-举例

❑ 某多媒体信息查询系统—专家估算

❑ 专家1: 1, 8, 9 $\Rightarrow (1+9+4 * 8) / 6 = 7$ (万元)

❑ 专家2: 4, 6, 8 $\Rightarrow (4+8+4*6) / 6 = 6$ (万元)

❑ 估算结果 $= (6+7) / 2 = 6.5$ (万元)

估算方法总结

1. 代码行估算法
2. 功能点估算法
3. 用例点估算法
4. 类比（自顶向下）估算法
5. 自下而上估算法
6. 参数估算法
7. 专家估算法

6.3.1 实用软件估算步骤

是一种自下而上和参数法的结合模型, 步骤如下:

1. 对任务进行分解: $1, 2, \dots, l, \dots, n$
2. 估算每个工作包的成本 E_i
3. 直接成本 $= E_1 + E_2 + \dots + E_i + \dots + E_n$
4. 间接成本估算
5. 项目总估算成本= 直接成本+间接成本

6.3.1-1 估算每个任务的成本

- ❑ 直接估算成本 E_i
- ❑ 先估算规模 Q_i ，然后估算成本 $E_i = Q_i * \text{人力成本参数}$



6.3.1-2 直接成本估算

- 直接成本组成

- 开发成本

- 管理成本

- 质量成本

- 例如：人力成本参数=5万/人月，30人月(包括开发\管理\质量)规模的项目的直接成本是150万

直接成本估算—简易估算：

开发（工作量）规模：

Scale (Dev)（单位：人月）

管理、质量（工作量）规模：

Scale (Mgn) = $a * \text{Scale (Dev)}$

[a为比例系数：例如：20%—25%]

直接成本 = $\text{Scale (Dev)} + a * \text{Scale (Dev)}$



6.3.1-3 间接成本

间接成本估算：

1. 按照企业模型直接估算：
2. 简易算法：
 - $\text{间接成本} = \text{直接成本} \times \text{间接成本系数}$
 - 例如：间接成本系数=0.3

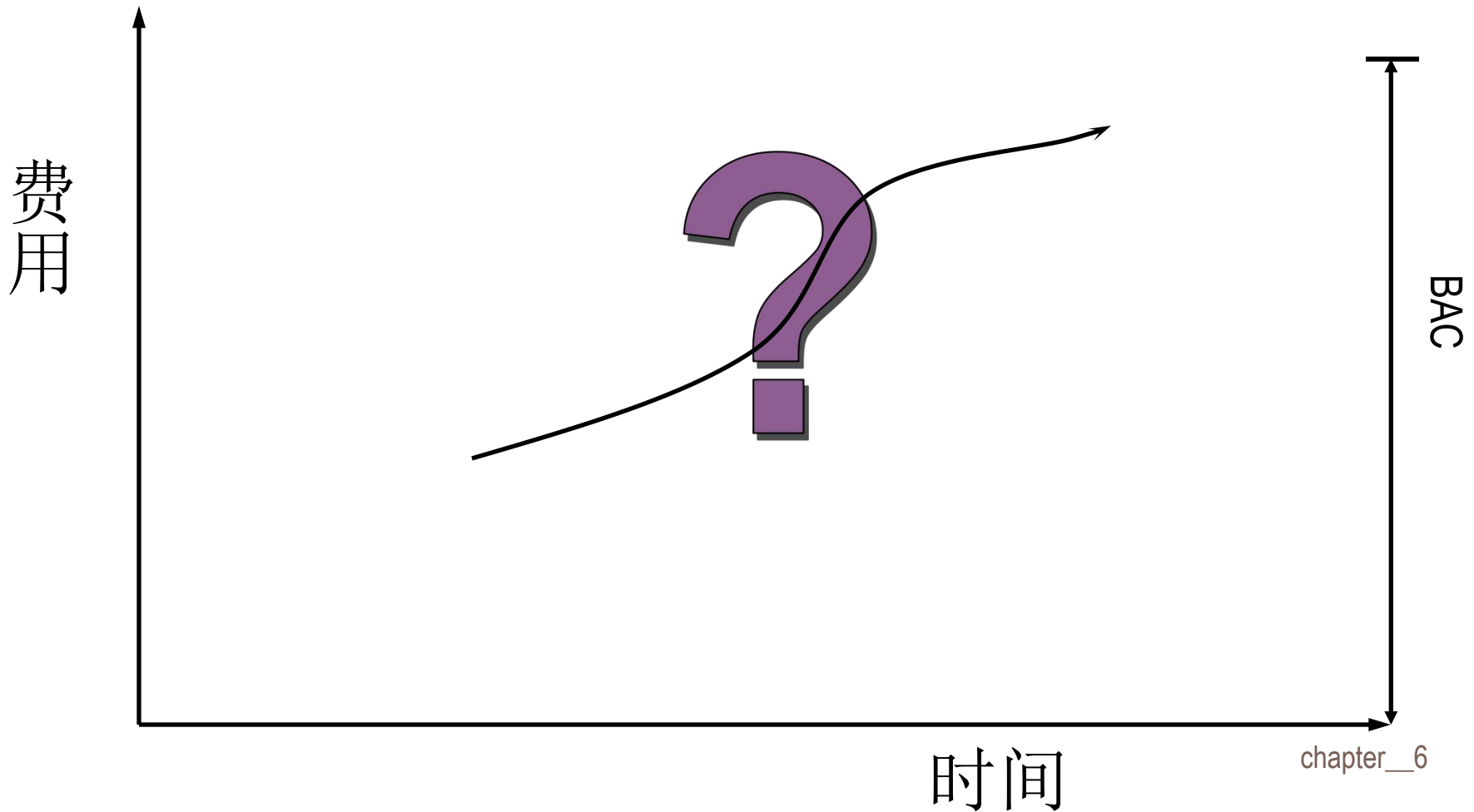


6.3.1-3 项目总估算成本

- 估算成本=直接成本+间接成本
 - 估算成本=直接成本+直接成本*间接成本系数
 - 估算成本=直接成本(1+间接成本系数)
 - 估算成本=规模*人力成本参数(1+间接成本系数)
 - 成本系数=人力成本参数*(1+间接成本系数)
-
- 简易算法:
 - 估算成本=规模*成本系数
 - 例如：成本系数= 8万/人月



总估算成本 (BAC)



本章要点

- ❑ 一、估算过程概念
- ❑ 二、估算方法
- ❑ 三、成本预算
- ❑ 四、案例分析
- ❑ 五、课程实践

6.4.1 成本预算

- 成本预算是将项目的总成本按照项目的进度分摊到各个工作单元中去
- 成本预算的目的是产生成本基线

6.4.1-0 项目成本预算

分配项目成本预算包括三种情况：

1. 给任务分配资源成本
2. 给任务分配固定资源成本
3. 给任务分配固定成本

6.4.1-1 给任务分配资源成本

- ❑ 与资源的基本费率紧密相连
- ❑ 设置资源费率
 - ❑ 标准费率
 - ❑ 加班费率
 - ❑ 每次使用费率
 - ❑ 。 。 。 。 。 。

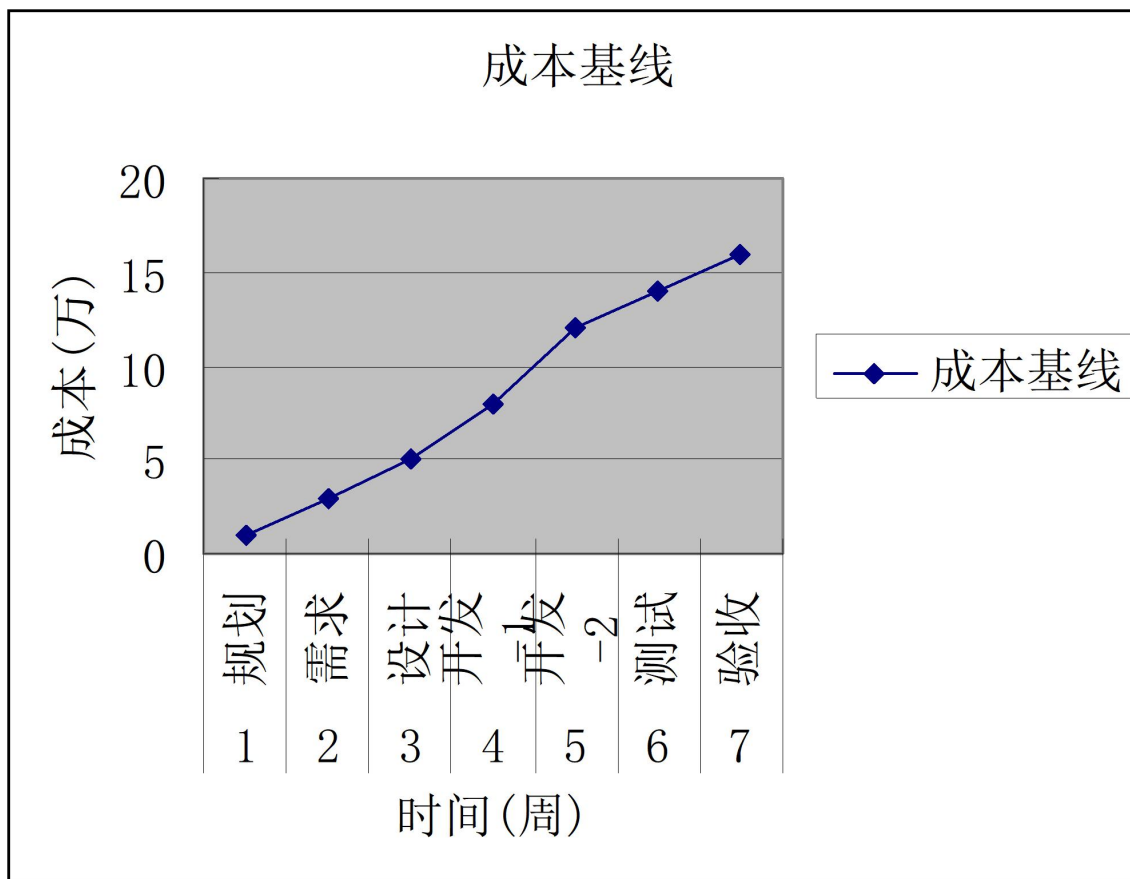
6.4.1-2 分配固定资源成本

- ❑ 当一个项目的资源需要固定数量的资金时，可以向任务分配固定资源成本。
- ❑ 例如：项目中的一个兼职人员成本

6.4.1-3 分配固定成本

- ❑ 有些任务是固定成本的类型的任务，也就是说，管理者知道某项任务的成本不变，不管任务的工期有多长，或不管任务使用了那些资源。在这种情况下，管理者向任务直接分配成本。
- ❑ 例如：某外包任务、培训任务

6.4.1-4 成本基线



本章要点

- ❑ 一、估算过程概念
- ❑ 二、估算方法
- ❑ 三、成本预算
- ❑ 四、案例分析
- ❑ 五、课程实践

6.5.1 医疗信息商务平台成本估算

MED的2个估算方法：

- 自下而上的估算
- 用例点估算

MED自下而上的估算

表 6A-5：自下而上的估算

医疗信息商务平台		人天	小计	总计
F1: 用户				396
	F1.1: 注册		15	
		个人注册	4	
		组织注册	4	
		协会/学会注册	4	
		登录	3	
	F1.2: 管理		26	
		用户信息	3	
		用户权限	8	
		统计分析	15	
F2: 产品信息				
	F2.1: 编辑		77	
		离线产品	23	
		在线产品	25	
		产品状态管理	29	
	F2.2: 浏览		27	
		按厂商浏览	12	
		按产品分类浏览	6	
		按医院科别浏览	9	
	F2.3: 查找		14	
F3: 网上交易				
	3.1 售前		24	
		3.1.1 客户的分级优惠	12	
		3.1.2 购买数量的优惠	12	
	3.2 售中		30	
		3.2.1 询价	15	
		3.2.2 接受订单	15	
	3.3 售后		40	
		3.3.1 统计分析	20	
		3.3.2 查看采购记录	20	
F4: 分类广告			50	
	4.1 与产品有关		9	
	4.2 与产品无关		10	
	4.3 匹配		15	
	4.4 招标		16	

F5: 协会/学会				33	
	5.1 编辑		8		
	5.2 浏览		9		
	5.3 管理		16		
F6: 医院管理				49	
	6.1 编辑		9		
	6.2 浏览		8		
	6.3 检索		16		
	6.4 管理		16		
F7: Email管理(购买)		2.4 万	1	1	
F8: Chat管理(现成)			2	2	
F9: 护士排班表(移植)			5	5	
F10: 联机帮助			3	3	

计算开发成本

1. 通过自下而上的计算，得知项目开发规模是**396**人天，开发人员成本参数=**1000**元/天，则内部的开发成本=**1000**元/天***396**天=**39.6**万元
2. 加上外包部分软件成本**2.4**万元，则开发成本=**39.6**万+**2.4**万=**42**万元

计算管理成本

针对本项目，管理成本=开发成本*10%。

所以，管理成本为=42万元*10%=4.2万元。

计算直接成本

因为，直接成本=开发成本+管理成本

所以，直接成本=**42万元+4.2=46.2万元**，

计算间接成本

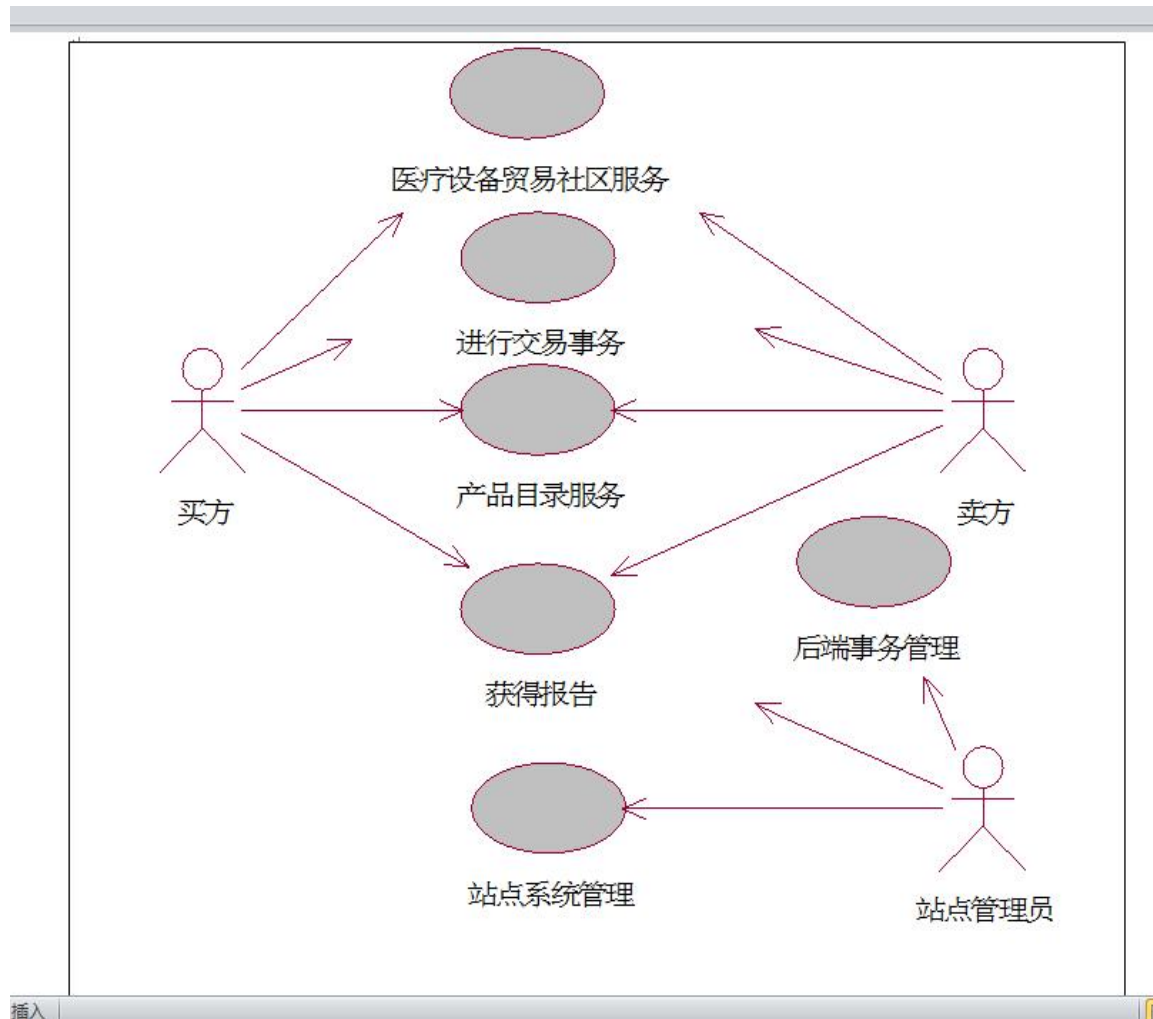
因为，间接成本=直接成本*20%

所以，间接成本=46.2万元*20%=9.24万元

计算总估算成本

项目总估算成本=直接成本+间接成本=
46.2万元+9.24万元=55.44万元。

MED用例点估算



角色：User

表 4-1 用户类型及角色表

用户类型	用户子类	角色	子角色
Medeal.com		Webmaster	
		内容管理经理	
		市场部经理	
组织	交易组织	厂商	管理者
			内容管理经理
			内容管理者
			销售经理
			销售人员
			决策人
			市场分析师
			合同履行人员
			售后服务经理
			售后服务人员
			一般人员
		经销商	管理者
			内容管理经理
			内容管理者
			销售经理
			销售人员
			决策人
			市场分析师
			合同履行人员
			售后服务经理
			售后服务人员
			采购经理
			采购员
			一般人员
		医院	管理者
			决策人
			采购经理
			采购员
			护士长
			护士
			一般人员
	非交易组织	协会/学会	管理者
			编辑者
			会员
个人		成员	
		非成员	

计算未调整的角色权值：UAW

$$UAW=18$$

表6A-1 UAW计算过程

序号	Actor 复杂度级别	权值	参与角色 Actor 数	UAWi
1	simple	1	1	1
2	average	2	7	14
3	complex	3	1	3
总计				18

计算未调整的用例权值：UUCW

UUCW=240

表6A-2 UUCW计算过程

序号	Use case 复杂度级别	权值	Use case 数量	UUCWi
1	simple	5	15	75
2	average	10	12	120
3	complex	15	3	45
总计				240

计算未调整的用例点：UUCP

$$UUCP = UAW + UUCW = 18 + 240 = 258$$

技术复杂度因子TCF

表6A-3 技术复杂度因子的定义

序号	技术因子	权值	Value 值	TCFi
1	TCF1	2.0	3	6.0
2	TCF2	1.0	5	5.0
3	TCF3	1.0	3	3.0
4	TCF4	1.0	5	5.0
5	TCF5	1.0	3	3.0
6	TCF6	0.5	3	1.5
7	TCF7	0.5	5	2.5
8	TCF8	2.0	3	6.0
9	TCF9	1.0	5	5.0
10	TCF10	1.0	3	3.0
11	TCF11	1.0	5	5.0
12	TCF12	1.0	3	3.0
13	TCF13	1.0	0	0.0
	TCF	$0.6 + (0.01 \times \sum TCF_i) = 1.08$		

环境因子ECF

表6A-4 环境因子的定义

序号	环境因子	权值	Value 值	<u>ECFi</u>
1	ECF1	1.5	3	4.5
2	ECF2	0.5	3	1.5
3	ECF3	1.0	3	3.0
4	ECF4	0.5	5	2.5
5	ECF5	1.0	3	3.0
6	ECF6	2.0	3	6.0
7	ECF7	1.0	0	0.0
8	ECF8	1.0	0	0.0
	ECF	$1.4 + (-0.03 \times \sum \text{ECFi}) = 0.785$		

计算用例点UCP

$$\begin{aligned} \text{UCP} &= \text{UUCP} \times \text{TCF} \times \text{ECF} = 258 \times 1.08 \times 0.785 \\ &= 218.7 \end{aligned}$$

规模： Effort

如果： PF=20工时/用例点

则： $\text{Effort} = \text{UCP} \times \text{PF} = 218.7 \times 20 = 4374$ 工时。

因为1人天=8工时，则项目规模： $4374/8 = 547$ 人天

如果1000元/人天，则成本54.7万

本章要点

- ❑ 一、估算过程概念
- ❑ 二、估算方法
- ❑ 三、成本预算
- ❑ 四、案例分析
- ❑ 五、课程实践

6.5.1 课程实践五：项目成本估算

实践目的：掌握软件项目规模成本估算方法

实践要求：

1. 复习软件成本估算方法
2. 采用用例点方法估算**SPM**项目
3. 采用自下而上方法估算**SPM**项目
4. 选择**1**个团队课堂上讲述**SPM**项目的两个估算结果

小结

❑ 成本估算

1. 代码行估算法
2. 功能点估算法
3. 用例点估算法
4. 类比（自顶向下）估算法
5. 自下而上估算法
6. 参数估算法
7. 专家估算法

❑ 成本预算