COCOMO II模型
——基本模型

舒风笛

2006-09-25

中国科学院软件研究所
Institute of Software,Chinese Academy of Sciences

提纲

一、COCOMO II简介

二、模型定义

模型校准

应用实例

模型的扩展与发展趋势

提纲

一、*COCOMO II简介*

二、模型定义

一、COCOMO II 简介 (1)

Constructive Cost Model（COCOMO）构造性成本模型

- 最初由**Barry Boehm**博士于**1981**年提出，发表在软件工程经济学《***Software Engineering Economics***》一书中
- ***《Software Cost Estimation with COCOMO II》对COCOMO II进行了介绍***
- 一个计划和执行软件项目的目标成本模型，可以被用作成本估算和相关活动的一个框架

  可识别具有显著作用的生产力改进因素并预测其回报
- 用于软件工作量、成本和进度估算
- 目前最广泛使用、充分文档化和得到系统校准的估算模型

  USC会员 **Rational** 软件公司

一、COCOMO II 简介 (2)

两个底层信息模型：

（1） 用于描述软件项目的框架，包括过程模型、文化、涉众、方法、工具、开发团 队以及软件产品的规模或复杂性；

（2） 经验库，可从历史案例估算出项目可能需要的相关资源（工作量与时间）

COCOMO 81与 COCOMO II

后者是前者的改进：

（1）对现代过程、方法、工具和技术的适用性；

（2）提供更大、更恰当的现代案例数据库，改进了模型的适应性，可在更多领域及其项目环境中进行优化

一、COCOMO II 简介(3)

COCOMO II模型目标

- 准确的成本和进度估算
- 能方便地校准、裁剪或扩展，以适应特殊情况
- 提供细致、易理解的输入、输出和假设定义
- 构造性模型：

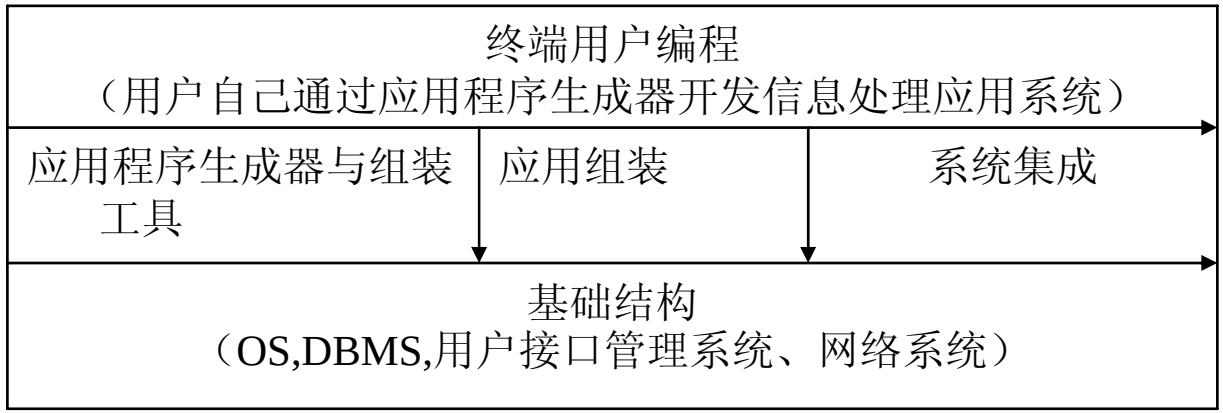
  帮助人们更好地理解所估算项目的本质
- 标准化模型

  需要数据校准
- 发展模型

一、COCOMO II 简介 (4)

未来软件从业市场模型

1994年Barry Boehm提出，指导COCOMO的构造



用户编程部分：　简单的基于活动估算即可；

用于应用组装部分的COCOMOII模型：基于对象点；

应用程序生成器、系统集成、基础结构：　基于应用组装模型（应用于早期原型开发工作量估算）＋更为详细的估算模型（应用于生命周期后期阶段）

一、COCOMO II 简介 (5)

系列模型：

- 应用组合（**Applications Composition)**模型
- 早期设计**(Early Design)**模型
- 后体系结构**(Post-Architecture)**模型

一、COCOMO II 简介(6)

应用组合模型

- 应用于早期原型开发工作量估算
- 最早阶段、快速开发或螺旋周期所涉及的原型开发活动，以解决潜在的高风险问题，如用户界面，软件/系统交互，性能或者技术成熟度；
- 以对象点（应用点，**application points** ）来衡量规模（带权的屏幕元素、报告、**3GL**模块数），每个应用点都有权重，由一个简单、中等、困难三级的复杂性因子来表示

一、COCOMO II 简介(7)

早期设计模型

- 涉及探索软件/系统体系结构的可选方案或增量开发策略
- 使用功能点描述操作概念，包括7个成本驱动因子

后体系结构模型

- 准备开发并支持一个实际系统，已有一个生命周期体系结构
- 涉及一个软件产品的实际开发和维护，通常以源代码和/或功能点来衡量规模，采用17个工作量乘数，5个项目的指数比例因子

一、COCOMO II 简介(8)

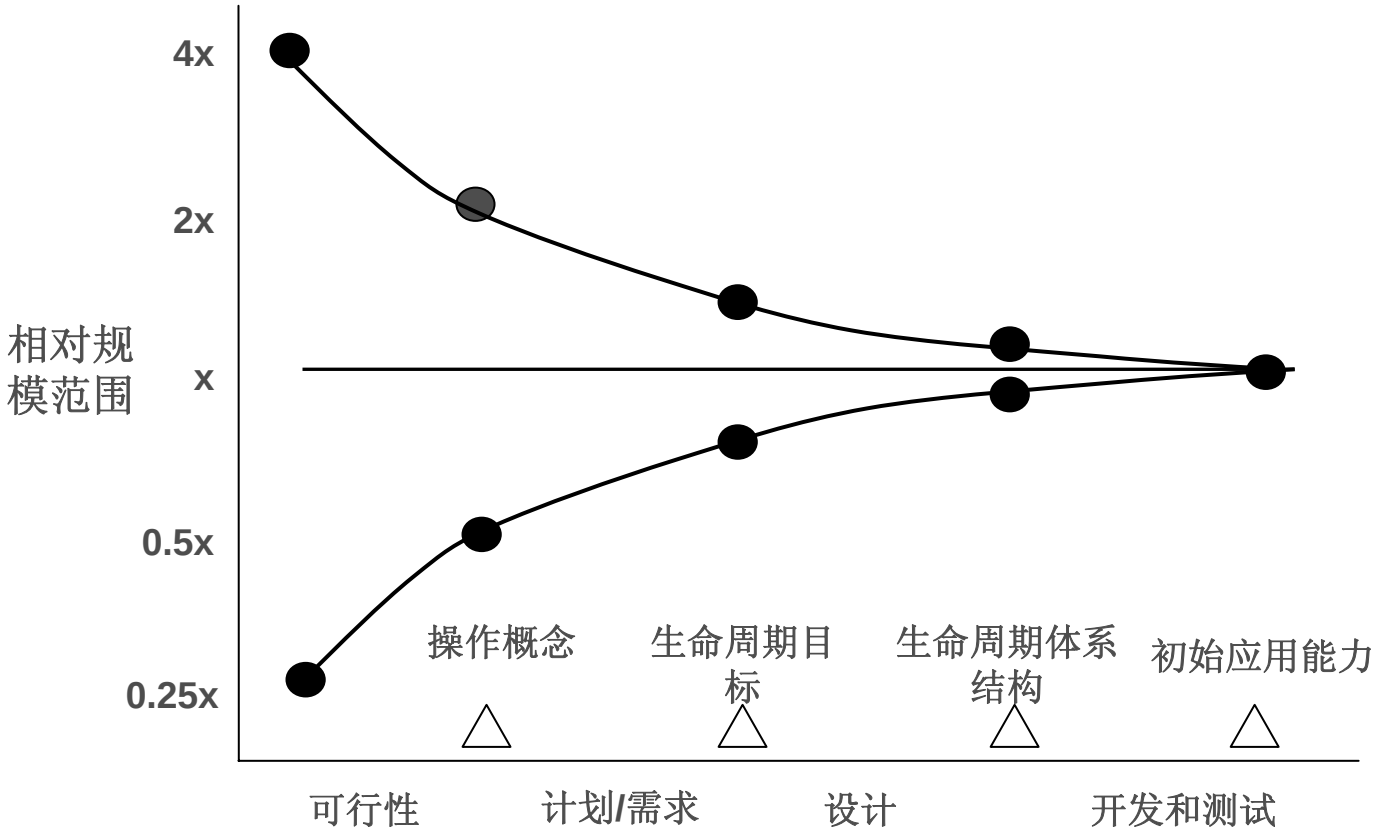
模型可裁剪组合原理所基于的假设：

1. 现在和未来的软件项目将按特有的过程驱动因子来裁剪其过程，而不是单一的瀑布模型；
2. 所用软件成本估算模型的粒度需求与支持软件成本估算的可用信息的粒度一致；
3. **COCOMO II**能在项目早期提供粗粒度的成本驱动因子信息，在后期逐步提供更详细的信息，产生依赖于估算输入定义程度的范围估算

一、COCOMO II 简介(9)

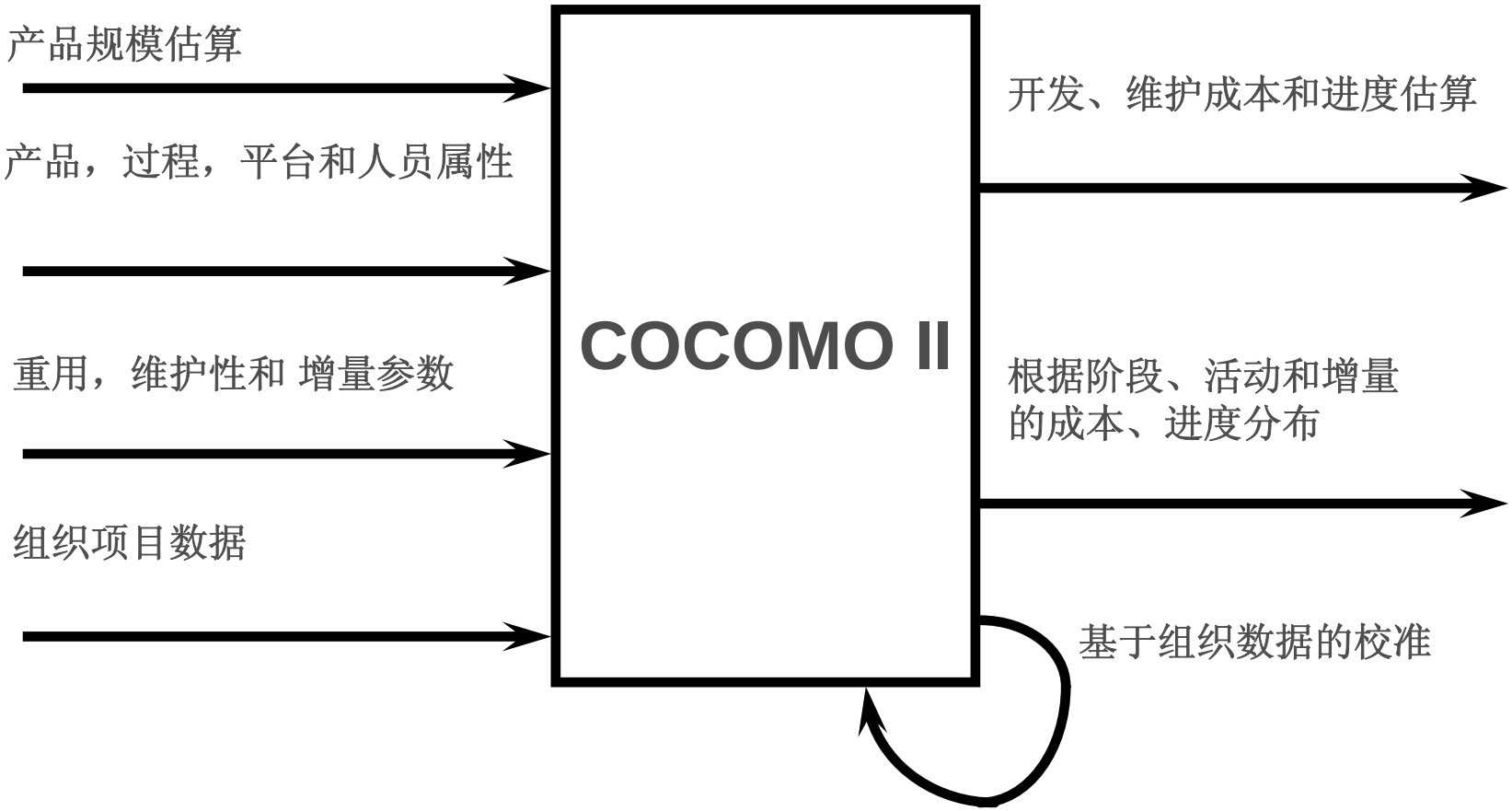
软件估算精确度

不确定性变动范围可用作估算变动范围的出发点



一、COCOMO II 简介(10)

COCOMO 黑盒模型



一、COCOMO II 简介(11)

COCOMO 在软件决策方面的应用

- ✓ 投资决策和商业案例分析
- ✓ 建立项目预算和进度
- ✓ 权衡分析
- ✓ 成本风险管理
- ✓ 开发与重用决策
- ✓ 遗留软件逐步淘汰决策
- ✓ 软件重用和产品线决策
- ✓ 过程改进决策

提纲

一、COCOMO II简介

二、*COCOMO II模型定义*

2.1 规模估算

2.2 工作量估算

2.3 进度估算

2.4 软件维护

2.1 规模估算

只采用影响工作量的规模数据：新代码和拷贝并经修改的代码

2.1.1 源代码行（**SLOC**）计算

2.1.2 未调整功能点（**UFP**）计算

2.1.3 **UFP**与**SLOC**关联

2.1.4 累加新的、改编的和复用的代码

2.1.5 需求演化和易变性（***REVL***）

2.1.6 自动转换的代码

2.1.7 软件维护的规模

2.1.1 源代码行（SLOC）计算

- 历史数据或专家意见 (可能的，最不可能的，最可能的规模)
- 规模表示为源代码千行数（KSLOC）
- 源代码一般不包括未交付的支持软件，如测试驱动程序；但若这些不交付支持软件的开发与交付软件的开发同样仔细，则需要计算在内，目的是度量投入程序开发中的智力工作量
- COCOMO选择逻辑源语句作为标准的代码行，可使用逻辑源语句定义检查表

2.1.2 未调整功能点（UFP）计算 (1)

- 功能点成本估算方法基于软件项目中的功能数和一组独立的项目因子；
- 基于项目生命周期早期就可得到的信息；
- 通过量化与主要外部数据或控制输入、输出或文件类型相联系的信息处理功能来度量软件项目

2.1.2未调整功能点（UFP）计算 — COCOMO II使用的功能点规模度量(1)

区分以下5种用户功能类型；每个功能类型实例再按照复杂等级分类，复杂等级确定一组权重，应用到相应的功能数上确定UFP数——

功能点	描述
外部输入（External Input, EI）	进入被度量软件系统外部边界的每一种唯一的用户数据或用户控制输入类型
外部输出（External Output, EO）	从被度量软件系统的外部边界出来的每一种唯一的用户数据或用户控制输出类型
内部逻辑文件（Internal Logical File, ILF）	把软件系统中主要的用户数据或控制信息逻辑组，计算为一个逻辑的内部文件类型。包含软件系统产生的、使用的或维护的每个逻辑文件
外部接口文件（External Interface File， EIF）	软件系统间传递或共享的文件应被每一个系统计算为外部接口文件类型
外部查询（External Inquiry, EQ）	每一种唯一的输入输出组合，此时输入引起并产生一个直接的输出，因此计算为一个外部查询类型

2.1.2未调整功能点（UFP）计算 — COCOMO II使用的功能点规模度量(2)

用于确定未调整功能点的**COCOMO II**过程

（1）按类型确定功能数；

（2）确定复杂性等级；

（3）应用复杂性权重；

（4）计算未调整功能点数：所有带权重的功能计数相加

- 遵循**IFPUG(International Function Point Users Group)**所给定义
- 可用于早期设计和后体系结构模型中

2.1.3 规模估算 — UFP与SLOC关联

COCOMO II采用向后细化（backfining）表把UFP转换为等价的SLOC

q1

2.1.4 累加新的、改编的和复用的代码 (1)

- 复用代码：事先存在的以“黑盒”形式插入产品中的代码；
- 改编代码：事先以“白盒”形式存在、经改编后为产品所用的代码
- 把复用代码和改编代码的有效规模调整为等价的新代码行：等价源代码行**ESLOC**
- **COCOMO II**采用非线性估算模型处理软件复用

q1

AAM公式要改

qinzhongsen, 2006-9-23

2.1.4 累加新的、改编的和复用的代码 (2)

$$\text{等价}KSLOC = \text{改编}KSLOC \times (1 - \frac{AT}{100}) \times AAM$$

AT：附加因子，重构时自动转换代码的百分比；

AAM： **Adaptation Adjustment Modifier** 改编调整修改量

$$AAM = \begin{cases} \dfrac{AA + AAF \times (1 + [0.02 \times SU \times UNFM])}{100}, & \text{当} AAF \le 50\text{时} \\ \dfrac{AA + AAF + (SU \times UNFM)}{100}, & \text{当} AAF > 50\text{时} \end{cases}$$

SU: 软件理解增量； **UNFM:** 程序员不熟悉性； **AA**：评估与消化

AAF：**Adaptation Adjustment Factor** 改编调整系数

$$AAF = (0.4 \times DM) + (0.3 \times CM) + (0.3 \times IM)$$

DM: 设计修改百分比，修改是为了使它适应新的目标和环境（主观量）

CM：代码修改百分比

IM：集成改编或复用软件所需的集成工作量的百分比

2.1.4 累加新的、改编的和复用的代码 (3)

软件理解增量SU的等级量表

若无DM和CM（不加修改地使用组件，则不需要SU）

	很低	低	标称	高	很高
结构	低内聚,高耦合,面条式代码	偏低内聚,高耦合	结构十分良好,存在一些薄弱环节	高内聚,低耦合	模块性很强,信息隐藏于数据/控制结构中
应用清晰	程序与应用不匹配	程序与应用有某种相关	程序与应用中度相关	程序与应用高度相关	程序与应用清晰匹配
自描述	代码难于理解,文档丢失/难于理解或陈旧	存在一些代码注释和标题,有一些有价值的文档	中等水平的代码注释、标题、文档	较好的代码注释和标题，有用的文档，但存在一些薄弱环节	代码能自描述,文档是最新的,组织良好,并包含设计原理
对 ESLOC 的 SU 增量	50	40	30	20	10

2.1.4 累加新的、改编的和复用的代码 (4)

- 用于评估与消化增量（AA）的等级量表

AA 增量	AA 工作量等级
0	无
2	基本的模块搜索和文档化
4	一些模型测试和评估（T&E），文档化
6	相当多的模块 T&E, 文档化
8	广泛的模块 T&E, 文档化

2.1.4 累加新的、改编的和复用的代码 (5)

- 程序员不熟悉性（**UNFM**）的等级量表

UNFM 增量	不熟悉性等级
0.0	完全熟悉
0.2	大部分熟悉
0.4	部分熟悉
0.6	有点熟悉
0.8	大部分不熟悉
1.0	完全不熟悉

2.1.4 累加新的、改编的和复用的代码 (6)

改编软件的参数约束和指导原则

代码种类	复用参数					
	DM	**CM**	**IM**	**AA**	**SU**	**UNFM**
新 - 所有原来的软件	无有效值					
改编 -改变先存在的软件	0% - 100% 一般应用 > 0%	0^{+}% - 100% 通常> DM 且必须是 > 0%	0% - 100+% IM 通常到中值且可能 > 100%	0% – 8%	0% - 50%	0 - 1
复用 - 不改变现有软件	0%	0%	0% - 100% 很少为 0%, 但可能非常小	0% – 8%	无有效值	
COTS - 成品软件（常需要新的连接代码来包装 COTS）	0%	0%	0% - 100%	0% – 8%	无有效值	

2.1.5 需求演化和易变性(*REVL*)

- 使用因需求演化而遗弃代码的百分比**REVL**因子调整由需求演化和易变性（由任务或用户接口变化、技术更新等）引起的产品有效规模

如：

一个项目交付了**10**万条指令，但遗弃了**2**万条，则**REVL**为**20**。在**COCOMO II**估算中，项目的有效规模为**12**万条指令。

$$Size = \left(1 + \frac{REVL}{100}\right) \times \text{Size}_D$$

2.1.6自动转换的代码

依赖于自动工具的效率

- **AT**：重构时自动转换代码的百分比；
- **ATPROD**：自动转换生产率

$$PM_{Auto} = \frac{\text{改编的}SLOC \times (\frac{AT}{100})}{ATPROD}$$

2.1.7 软件维护的规模

计算软件维护的规模

（Size）$_M$=[（基本代码规模）×MCF]×MAF

若只对现有基本代码做少量增补或修改时：

（Size）M＝（增加规模＋修改规模）×MAF

MAF：维护调整系数 $MAF = 1 + (\frac{SU}{100} \times UNFM)$

MCF：维护改变系数 $MCF = \frac{\text{增加规模} + \text{修改规模}}{\text{基本代码规模}}$

提纲

一、COCOMO II简介

二、*COCOMO II模型定义*

2.1 规模估算

2.2 工作量估算

2.3 进度估算

2.4 软件维护

2.2 工作量估算

2.2.1 工作量估算模型

2.2.2 比例因子

2.2.3 成本驱动因子（工作量乘数）

2.2.4 多模块的工作量估算

2.2 工作量估算

- **COCOMO II**中，工作量用人月（**person-month, *PM***）表示，一个人在一个月内从事软件开发项目的时间数；
- 每人月的人时数（ **person-hours per person-month**，***PH/PM***）作为可调整系数，标称值为**152**小时***/PM***，不包括节假日、周末
- 开发进度或开发时间（**Time to Develop**，***TDEV***）：完成项目要花费的时间

2.2.1工作量估算模型（1）

$$PM = A \times Size^{E} \times \prod_{i=1}^{\text{成本驱动因子数}} EM_i + PM_{Auto}$$

$$E = B + 0.01 \times \sum_{j=1}^{5} SF_j \qquad PM_{Auto} = \frac{\text{改编的}SLOC \times (\frac{AT}{100})}{ATPROD}$$

其中:

- *A*：从历史项目数据推导出的常量 (COCOMOII.2000中*A* = 2.94)
- B为可以校准的常数， COCOMOII.2000中B = 0.91
- *Size*：表示为源代码千行数，也可从功能点或者对象点转换得到
- *E*：指数，依赖于5个比例因子的总和
- SF_i：比例因子
- EM_i 为第i个成本驱动因子的工作量乘数
- AT:通过自动转换来重构改编的SLOC之百分比
- ATPROD: 自动转换生产率

2.2.1 工作量估算模型（2）

- 早期设计模型与后体系结构模型中，**A,B,C,D,SF1,…,SF5**的值相同
- 早期设计模型中，**EM**个数为**6**，是后体系结构模型的**17**个对应值进行合并获得的
- PM_{NS}表示改公式为标称进度下的工作量，进度压缩或延长的影响是由成本驱动因子***SCED***（要求的开发进度）来涵盖
- **COCOMO II**模型可用于整个项目或多个模块组成的项目估算工作量和进度，除**SCED**和比例因子外，每个模块的规模和其他成本驱动因子的等级可不同

2.2.2 比例因子（1）

只用于整个项目

$$E = B + 0.01 \times \sum_{j=1}^{5} SF_j$$

E<1.0 项目表现出规模经济（规模翻倍，工作量不至于翻倍，生产率随产品规模的增加而提高）

E=1.0 线性模型，规模经济性与不经济性平衡，常用于小项目的成本估算

E>1.0 规模不经济性，由于：人员间交流开销和大型系统集成开销的增长

2.2.2 比例因子（2）

比例因子**SF**

- 项目工作量指数变化或生产率变化的重要成因
- 有一个从“很低”到“极高”的等级变动范围，每个等级有一个权重，权重的值称为比例因子（***SF***）

五个比例因子：

- ✓ 先验性 **(PREC)**
- ✓ 开发灵活性 **(FLEX)**
- ✓ 体系结构/风险化解 **(RESL)**
- ✓ 团队凝聚力 **(TEAM)**
- ✓ 过程成熟度 **(PMAT)**

2.2.2 比例因子（3）

先验性 (PREC)

目标系统新的程度和过去经验可应用的程度，如果产品与以前开发过的项目类似，则先例性高

特征	很低	标称/高	极高
对产品目标有组织级的理解	一般	相当多	完全
相关软件系统的工作经验	中等	相当多	广泛
与相关新硬件和操作程序的协同开发	广泛	中等	一些
需要创新的数据处理体系结构和算法	相当多	一些	极少

2.2.2 比例因子（4）

开发灵活性 (FLEX)

衡量软件产品必须与外部需求与接口达成一致的程度

特征	很低	标称/高	极高
软件性能与已建立的需求一致	完全	相当	基本
软件性能与外部接口规范一致	完全	相当	基本
以上两点均一致，另外鼓励及早完成	高	中等	低

2.2.2 比例因子（5）

体系结构/风险化解 (RESL)

度量通过产品设计评审(PDR)得到的设计彻底性和通过PDR得到的风险消除(设计的完全性和相应的风险消除程度)

特征	很低	低	标称	高	很高	极高
通过风险管理计划找出所有的关键风险项，通过 PDR 或生命周期体系结构（Life Cycle Architecture，LCA）建立解决这些风险的里程碑	无	极少	一些	一般	大多数	完全
通过与风险管理计划相一致的 PDR 或 LCA，得到进度、预算和内部里程碑	无	极少	一些	一般	大多数	完全
给定总的产品目标时，建立体系结构所用的开发进度百分比	5	10	17	25	33	40
项目所需的顶级软件体系结构的百分率	20	40	60	80	100	120
化解风险项、开发和验证体系结构说明的可用工具支持	无	极少	一些	良好	强大	完全
关键体系结构驱动因子中的不确定性级别：任务、用户界面、COTS、硬件、技术、性能	极端	显著	相当高	有些	少	极少
风险项的数目和危险程度	> 10 Critical	5-10 Critical	2-4 Critical	1 Critical	> 5 Non-Critical	< 5 Non-Critical

2.2.2 比例因子（6）

团队凝聚力 (TEAM)

- 协调涉众和最小化冲突的需要程度,是下列特征的主观加权平均值，以说明项目混乱和熵的原因
- 混乱与熵是因为难以协调项目涉众
- 涉众包括：用户、客户、开发者、维护者、协调人员及其他

特征	极低	低	标称	高	很高	极高
涉众的目标和文化一致	少	一些	基本	相当高	强烈	完全
一些涉众适应其他涉众的目标的能力和意愿	少	一些	基本	相当高	强烈	完全
涉众作为团队工作的经验	无	少	少	基本	相当多	广泛
为达成共同观点和承诺，涉众所进行的团队建设	无	少	少	基本	相当多	广泛

2.2.2 比例因子（7）

- 过程成熟度 (PMAT)

  围绕美国卡耐基梅隆大学软件工程研究所（SEI）的能力成熟度模型（CMM）而组织的，确定开发过程成熟度等级

  两种方法:

  （1）利用基于CMM的组织评估结果;

  （2）确定对CMM中的18个关键过程域（Key Process Area, KPA）的遵守百分比

2.2.2 比例因子（8）

关键过程域

等价过程成熟度级别EPML：确定对每个KPA的遵守百分比，综合所有18个KPA的目标判断而平均得到

$$EPML = 5 \times \left[\sum_{i=1}^{n} \left(\frac{KPA\%_i}{100} \right) \times \frac{1}{n} \right], \quad n<=18$$

“未使用”和“不知道”的项不计算

KPA	几乎总是 (>90%)	经常 (60-90%)	大约为一半 (40-60%)	偶尔 (10-40%)	即使有也很少 (<10%)	未使用	不知道
1 需求管理	❒	❒	❒	❒	❒	❒	❒
2 软件项目管理	❒	❒	❒	❒	❒	❒	❒
3 软件项目跟踪和监督	❒	❒	❒	❒	❒	❒	❒
4 软件子合同管理	❒	❒	❒	❒	❒	❒	❒

(参见 COCOMO II 模型定义手册)

2.2.2 比例因子（9）

- 用于COCOMO II面向的比例因子值SF_i

比例因子	很低	低	标称	高	很高	极高
PREC	全新的 6.20	绝大部分新的 4.96	有一些新的 3.72	基本熟悉 2.48	绝大部分熟悉 1.24	完全熟悉 0.00
FLEX	严格 5.07	偶尔放宽 4.05	放宽 3.04	基本一致 2.03	部分一致 1.01	通用目标 0.00
RESL	很少（20%） 7.07	一些（40%） 5.65	通常（60%） 4.24	一般（75%） 2.83	绝大多数（90%） 1.41	完全（100%） 0.00
TEAM	交流非常困难 5.48	交流有一些障碍 4.38	基本的交流协作 3.29	广泛地协作 2.19	高度协作 1.10	无缝协作 0.00
PMAT	SW-CMM1级的较低部分 7.80	SW-CMM1级的较高部分 6.24	SW-CMM2级 4.68	SW-CMM3级 3.12	SW-CMM4级 1.56	SW-CMM5级 0.00

----------或估算的等价过程成熟度等级（Equivalent Process Maturity Level, EPML）----------

2.2.2 比例因子（10）

- 用于获取影响完成项目所需工作量的软件开发特征；
- 一个成本驱动因子是一个模型系数；“驱动”由模型所估算的工作量；
- 每个因子有定性的等级，表示对开发工作量的影响程度；
- 当某个成本驱动因子的估计值处于给定等级之间，通常取接近标称的值

  如：处于高和极高之间，则取高
- 每个等级都有一个工作量乘数（**effort multiplier, EM**）值与其对应
- 早期设计模型采用**6**个工作量乘数，后体系结构模型采用**17**个工作量乘数（含**SCED**）
- 标称级别总是有**1.00**的工作量乘数，不改变所估算的工作量，非标称等级一般都改变工作量估算

2.2.3 成本驱动因子（1）

- 后体系结构成本驱动因子
- 早期设计成本驱动因子

2.2.3.1 后体系结构成本驱动因子（1）

- 产品因子**Product Factors**

  用于说明由正在开发产品的特征引起开发软件所需工作量的变化

  - 可靠性 **(RELY)**
  - 数据库规模 **(DATA)**
  - 产品复杂性 **(CPLX)**
  - 可重用性 **(RUSE)**
  - 文档编制 **(DOCU)**

- 平台因子

  平台指具有复杂硬件和基础软件的目标机（虚拟机）

  - 执行时间约束 **(TIME)**
  - 主存储约束**(STOR)**
  - 平台易变形 **(PVOL)**

2.2.3.1 后体系结构成本驱动因子（2）

人员因子 Personnel factors

- 按照开发团队的能力和经验进行分级
- 项目进行期间可能发生变化，反映经验的获取或人员出入项目组的状况
  - 分析员能力 (ACAP)
  - 程序员能力 (PCAP)
  - 应用经验 (APEX)
  - 平台经验 (PLEX)
  - 语言和工具经验 (LTEX)
  - 人员连续性 (PCON)

2.2.3.1 后体系结构成本驱动因子（3）

- 项目因子

  说明诸如现代软件工具的使用、开发组的地理位置和项目进度压缩等因素对工作量估算的影响

  - 软件工具(TOOL)
  - 多点开发 (SITE)
  - 要求的开发进度 (SCED)

2.2.3.1 后体系结构成本驱动因子 – 产品因子（1）

- 所要求的软件可靠性 **(RELY)**
  - 度量软件必须在一定时间内执行其预期功能的程度
  - 软件失效的影响程度?

RELY 描述	有点不方便	低，易弥补的损失	中等，易弥补的损失	高财务损失	性命攸关	
等级	很低	低	标称	高	很高	极高
工作量乘数	**0.82**	**0.92**	**1.00**	**1.10**	**1.26**	无

2.2.3.1 后体系结构成本驱动因子 – 产品因子（2）

数据库规模 (DATA)

- 获取大量测试数据的需求对产品开发的影响
- 通过计算“D（测试数据库的字节数）/P（程序中SLOC）”获得

DATA 描述		D/P<10	10<=D/P <100	100<=D/P <1000	D/P>=1000	
等级	很低	低	标称	高	很高	极高
工作量乘数	无	0.90	1.00	1.14	1.28	无

2.2.3.1 后体系结构成本驱动因子 – 产品因子（3）

- 产品复杂度(CPLX)
  - 分为5方面(区域):
    - 控制操作
    - 计算操作
    - 设备相关操作
    - 数据管理操作
    - 用户节目管理操作
  - 选择某一方面或者对多方面进行综合，以描述该产品或者产品子系统的特征

2.2.3.1 后体系结构成本驱动因子 – 产品因子（4）

- 可复用开发（**RUSE**）
  - 说明构造可在当前或未来项目中复用的组件所需的额外工作量（为使组件可重用而花费的额外工作量之可以重用）
  - 对项目的**RELY**和**DOCU**（见后）等级施加了约束：**RELY**等级最多比**RUSE**低一级

	很低	低	标称	高	很高	极高
RUSE 描述		无	跨项目	跨程序	跨产品线	跨多条产品线

2.2.3.1 后体系结构成本驱动因子 – 产品因子（5）

- 匹配生命周期的文档编制(DOCU)

  根据项目文档对其生命周期需求的适应性，衡量所要求的文档等级

	很低	低	标称	高	很高	极高
DOCU描述	未包含大量生命周期需求	未包含某些生命周期需求	符合生命周期需求规模	超出生命周期需求	大大超出生命周期需求	

2.2.3.1 后体系结构成本驱动因子 – 产品因子（6）

- 执行时间约束（**TIME**）
  - 度量强加到软件产品上执行时间的约束
  - 由系统或子系统预期消耗的执行时间资源与可用执行时间的百分比表示

	很低	低	标称	高	很高	极高
TIME 描述			使用不到 50%的可用存储空间	使用 70%的可用存储空间	使用 85%的可用存储空间	使 用 95%的可用存储空间

2.2.3.1 后体系结构成本驱动因子 – 产品因子（7）

主存储约束（STOR）

- 代表施加到软件产品或其子系统上的主存储约束的程度
- 许多应用的扩展使该因子仍有意义

	很低	低	标称	高	很高	极高
STOR 描述			使用不到 50% 的可用存储空间	使用70% 的可用存储空间	使用85% 的可用存储空间	使用95% 的可用存储空间

2.2.3.1 后体系结构成本驱动因子 – 产品因子（8）

平台易变性（PVOL）

度量软件产品调用的硬件和软件(OS、DBMS等)变化的频率

	很低	低	标称	高	很高	极高
PVOL 描述		每 12 个月有主要变化，每个月有次要变化	主要: 6 月；； 次要: 2 周	主要: 2 月; 次要: 1 周.	主要: 2 周; 次要: 2 天	

2.2.3.1 后体系结构成本驱动因子 – 人员因子（1）

分析员能力(ACAP)

- 考虑从事需求分析、高级设计和详细设计的人员的分析和设计能力、效率和彻底性以及交流和协作能力
- 不考虑经验级别

	很低	低	标称	高	很高	极高
ACAP 描述	第 15 个百分点	第 35 个百分点	第 55 个百分点	第 75 个百分点	第 90 个百分点	

2.2.3.1 后体系结构成本驱动因子 – 人员因子（2）

程序员能力(PCAP)

考虑程序员（作为小组）的能力、效率和彻底性以及交流和协作能力

	很低	低	标称	高	很高	极高
PCAP 描述	第 15 个百分点	第 35 个百分点	第 55 个百分点	第 75 个百分点	第 90 个百分点	

2.2.3.1 后体系结构成本驱动因子 – 人员因子（3）

人员连续性(PCON)

根据项目的年人员周转率确定

	很低	低	标称	高	很高	极高
PCON 描述	48% /年	24% /年	12% /年	6% /年	3% / 年	

应用经验(APEX)

衡量项目组开发软件系统或子系统的经验级别

	很低	低	标称	高	很高	极高
APEX 描述	<=2 个月	6 个月	1 年	3 年	6 年	

2.2.3.1 后体系结构成本驱动因子 – 人员因子（4）

平台经验(PLEX)

衡量项目组使用平台（包括图形用户界面、数据库、网络和分布式中间件等）的经验级别

	很低	低	标称	高	很高	极高
PLEX 描述	<=2 个月	6 个月	1 年	3 年	6 年	

语言和工具经验(LTEX)

衡量项目组用于开发软件系统或子系统的编程语言和软件工具的经验

2.2.3.1 后体系结构成本驱动因子 – 项目因子（1）

软件工具的使用(TOOL)

评定开发过程中软件工具的能力、成熟度和集成影响

	很低	低	标称	高	很高	极高
TOOL描述	编辑、编码、调试工具	简单的前端和后端CASE，很少集成	基本的生命周期工具，中度集成	健壮、成熟的生命周期工具，中度集成	健壮、成熟、主动的生命周期工具，与过程、方法、复用的良好集成	

2.2.3.1 后体系结构成本驱动因子 – 项目因子（2）

多点开发(SITE)

评定软件开发的地点分布和交流支持情况

	很低	低	标称	高	很高	极高
SITE:分布描述	国际	多城市和多企业	多城市或多企业	同一城市或大区域	同一建筑或企业	完全同处一地
SITE:交流描述	一些电话，邮件	专用电话，FAX	窄带电子邮件	宽带电子交流	宽带电子交流，偶尔视频会议	多媒体交互

2.2.3.1 后体系结构成本驱动因子 – 项目因子（3）

要求的开发进度(SCED)

- 度量施加在软件开发项目组上的进度约束
- 根据相对于需要一定工作量的项目的标称进度，所延长或加速的百分比来确定的
- 进度加速趋向于在早期阶段需要更多工作量以消除风险和细化体系结构，在后期阶段则需要更多工作量以并行地完成更多的测试和文档工作
- 唯一用于描述整个项目进度伸缩的成本驱动因子

	很低	低	标称	高	很高	极高
SCED描述	标称进度的 75%	标称进度的85%	标称进度的 100%	标称进度的130%	标称进度的160%	

2.2.3.2 早期设计模型成本驱动因子 (1)

通过组合后体系结构模型的成本驱动因子得到早期设计成本驱动因子

早期设计的成本驱动因子	对应后体系结构的成本驱动因子的组合
产品可靠性与复杂性（RCPX）	RELY, DATA, CPLX, DOCU
可复用性（RUSE）	RUSE
平台难度（PDIF）	TIME, STOR, PVOL
人员能力（PERS）	ACAP, PCAP, PCON
人员经验（PREX）	APEX, PLEX, LTEX
设施（FCIL）	TOOL, SITE
要求的开发速度（SCED）	SCED

2.2.3.2 早期设计模型成本驱动因子 (2)

映射方法

（1）将后体系结构成本驱动因子的等级分别对应一个数值等级

很低：1； 低：2；标称：3； 高：4；很高：5，极高：6

（2）求后体系结构中所有成本驱动因子的数值和；

（3）将总和分配到扩展的早期设计模型从“极低”到“极高”的等级量表

注：标称总和＝相应后体系结构组成元素的标称等级之和

2.2.3.2 早期设计模型成本驱动因子 (3)

产品可靠性与复杂性（RCPX）

对应： **RELY, DATA, CPLX, DOCU**

等级数值之和的变化范围：**5（VL, L, VL, VL）到21（VH, VH, EH, VH）**

RCPX描述							
RELY, DATA, CPLX, DOCU等级求和	**5，6**	**7，8**	**9 - 11**	**12**	**13 - 15**	**16 - 18**	**19-21**
强调可靠性、文档化	很少	少	有些	基本	强烈	很强烈	极强
产品复杂性	很简单	简单	有些	中等	复杂	很复杂	异常复杂
数据库大小	小	小	小	中等	大	很大	很大
等级	**极低**	**很低**	**低**	**标称**	**高**	**很高**	**极高**
工作量乘数	0.49	0.60	0.83	1.00	1.29	1.81	2.72

2.2.3.2 早期设计模型成本驱动因子 (4)

平台难度PDIF

PDIF描述					
TIME, STOR和PVOL等级之和	8	9	10 - 12	13 - 15	16, 17
时间和存储约束	<=50%	<=50%	65%	80%	90%
平台易变性	很稳定	稳定	有些不稳定	不稳定	非常不稳定
等级	低	标称	高	很高	极高
工作量乘数	0.87	1.00	1.29	1.81	2.61

2.2.3.2 早期设计模型成本驱动因子 (5)

人员能力（PERS）和映射示例

工作量乘数的推导方法：对于给定的早期设计等级，求与其相关的每个工作量乘数组合的乘积，再取其评价值

PERS描述							
ACAP, PCAP,PCON 等级求和	3,4	5,6	7,8	9	10, 11	12, 13	14, 15
ACAPHE PCAP百分比	20%	35%	45%	55%	65%	75%	85%
年人员流动率	45%	30%	20%	12%	9%	6%	4%
等级	极低	很低	低	标称	高	很高	极高
工作量乘数	2.12	1.62	1.26	1.00	0.83	0.63	0.50

2.2.3.2 早期设计模型成本驱动因子 (6)

人员经验(PREX)

PREX描述							
APEX, PLEX和LTEX等级之和	3,4	5,6	7,8	9	10, 11	12, 13	14, 15
应用、平台、语言和工具经验	<=3个月	5个月	9个月	1年	2年	4年	6年
等级	极低	很低	低	标称	高	很高	极高
工作量乘数	2.12	1.62	1.26	1.00	0.83	0.63	0.50

2.2.3.2 早期设计模型成本驱动因子 (7)

- 设施（*FCIL*）

FCIL描述							
TOOL和SITE等级之和	2	3	4，5	6	7，8	9，10	11
TOOL支持	少	一些	简单的CASE工具	基本生命周期工具	良好、中度集成	健壮、中度集成	健壮，良好集成
多点条件	对复杂多点开发（M/S）有少量的支持	对复杂M/S开发有些支持	对中等复杂的M/S开发有些支持	对中等复杂的M/S开发有基本支持	对中等复杂的M/S开发有强有力支持	对简单M/S开发有强有力支持	对同处一地或简单M/S开发有超强支持
等级	极低	很低	低	标称	高	很高	极高
工作量乘数	1.43	1.30	1.10	1.00	0.83	0.73	0.62

2.2 工作量估算

2.2.1 估算模型

2.2.2 比例因子

2.2.3 成本驱动因子（工作量乘数）

2.2.4 多模块的工作量估算

2.2.4 多模块的工作量估算 (1)

适用于一级子组件的系统估算 q3

（1）对所有组建的规模$Size_i$求和，得到总规模

$$Size_{Aggregate} = \sum_{i=1}^{n} Size_i$$

（2）将项目级驱动因子（比例因子和**SCED**）应用于总模型，导出基本总工作量PM_{Basic}

$$PM_{Basic} = A \times (Size_{Aggregate})^{E} \times SCED$$

q3

核实一级的概念

qinzhongsen, 2006-9-23

2.2.4 多模块的工作量估算 (2)

（3）基于每个组件对总规模的贡献，将基本工作量按比例分配给每个组件，据此确定每个组件的基本工作量PM_{Basic}（i）

$$PM_{Basic(i)} = PM_{Basic} \times (\frac{Size_i}{Size_{Aggregate}})$$

（4）将组件级的成本驱动因子（除SCED）应用于每个组件的基本工作量

$$PM_i = PM_{Basic(i)} \times \sum_{j=1}^{16} EM_j$$

2.2.4 多模块的工作量估算 (3)

（5）对每个组件的工作量求和，得到整个项目的总工作量$PM_{Aggregate}$

$$PM_{Aggregate} = \sum_{i=1}^{n} PM_i$$

（6）估算进度时，重复步骤2到5，但不应用第2步的SCED成本驱动因子。利用该修正的总体工作量导出进度

提纲

一、COCOMO II简介

二、*COCOMO II模型定义*

2.1 规模估算

2.2 工作量估算

2.3 进度估算

2.4 软件维护

2.3 进度估算模型

- 用于**COCOMO II** 早期设计和后体系结构阶段的初始进度公式：

$$TDEV(\text{月}) = [C \times (PM_{NS})^{(D+0.2\times(E-B))}] \times \frac{SCED\%}{100}$$

- **C**：可校准的**TDEV**系数，目前为**C=3.67**
- ***PM*$_{NS}$**:不考虑***SCED***工作量乘数时估算的***PM***
- **D**：可校准的**TDEV**规模基本指数；
- **SCED**％是**SCED**工作量乘数等级量表中的压缩/扩展百分比

提纲

一、COCOMO II简介

二、*COCOMO II模型定义*

2.1 规模估算

2.2 工作量估算

2.3 进度估算

2.4 软件维护

3.2.4 软件维护（1）

- 软件维护：

  在不改变现有软件基本功能的前提下，对其进行修改的过程

- **COCOMO II**模型假设软件维护成本一般具有与软件开发成本相同的成本驱动因子属性
- 维护包括：对原始产品小部分的重新设计和编码、接口的重新设计和开发，以及产品结构的细微修改
- 维护分为升级和修改两类
- 产品修改可进一步分为校正性、适应性或完善性

3.2.4 软件维护（2）

将COCOMO II应用于维护所需考虑的（1）：

- 不采用**SCED**（要求的开发进度）：

  维护周期通常是固定的时间

- 不采用**RUSE**（要求的可复用性）：

  维护一个组件的可复用性所需的额外工作量，基本上由所维护组件仔细的设计、文档编制和测试而减少的维护工作量所抵消

3.2.4 软件维护（3）

将COCOMO II应用于维护所需考虑的（2）：

- **RELY**（要求的软件可靠性）采用不同的工作量乘数，依赖于所要求的产品的可靠性：
  - ✓ 低可靠性要求下开发：需要更多工作量去修复潜在的错误；
  - ✓ 很高可靠性要求下开发：维护该可靠性级别所需工作量也高于标称值

RELY描述	少量的不方便	低，易于弥补的损失	中等，易于弥补的损失	高财务损失	性命攸关	
等级	很低	低	标称	高	很高	极高
工作量乘数	1.23	1.10	1.00	0.99	1.07	未定义

3.2.4 软件维护（4）

将COCOMO II应用于维护所需考虑的（3）：

- 比例因子***E***应用于更改了的**KSLOC**（增加、修改的，不包括删除的）
- 有效维护规模（$Size_M$）由维护调整系数***MAF***来调整遗留系统的影响

$$PM_M = A \times (Size_M)^E \times \prod_{i=1}^{15} EM_i$$

参考文献

- Barry W. Boehm, Software Engineering Economics, China Machine Press, ISBN 7-111-14389-2
- Barry W. Boehm, Software Cost Estimation with COCOMO Ⅱ, China Machine Press, ISBN 7-111-15777-X
- ……