COCOMO II—软件项目管理中的成本估算方法

周 杰，杜 磊
(同济大学 经济与管理学院，上海 200092)

摘 要：结构化成本模型(COCOMO: Costructive Cost Model)是一种成熟的软件成本估算方法，将介绍其第二版：COCOMO II模型的基本思路和使用方法，以及对原有COCOMO模型的改进。

关键词：COCOMO II；成本评估；软件项目管理
中图分类号：TP311.5 **文献标识码**：A **文章编号**：1001-3695(2000)11-0056-03

正如大多数工程项目一样，随着信息化工程的普及，企业用户认识到软件项目同样要进行项目管理和控制，要求软件开发商通过项目计划书估算成本，作出进度和质量保证，其中软件成本估算成为双方最有争议的首要问题。尤其是许多系统集成、网站开发项目，如果没有事先成本估算和对资金、时间、人力有效的管理和控制，绝大多数项目都会超支和延误进度，甚至项目失败。但是软件成本估算又是一个很难量化的过程，国内大多数项目的软件成本估算仅仅依靠经验和WBS (工作分解结构)法，这对关心过程改进的软件企业来说远远不够。南加州大学(USC)软件工程研究中心在COCOMO 81的基础之上，基于未来的软件生命周期过程，发表了COCOMO II模型，将DELPHI专家法与贝叶斯统计分析相结合，从模型中得到理想的成本估算结果。我们整理了有关资料，以前后两个COCOMO模型的比较为侧重点，介绍COCOMO II的理论和方法，希望能在国内软件项目管理中得到借鉴和使用。

1 COCOMO II的来历和发展

在过去的几十年间，出现了许多模型用于软件生命周期早期阶段的软件成本和进度估算。这些评估方法的目标就是为了保证在预算和质量要求的基础上，通过软件开发过程管理，及时交付软件产品。但是大多数的软件开发项目仍然超出了进度和成本。比较现有的成本模型，研究显示这些模型精确度并不令人满意。大多数模型都是专有的(如：SPR's Checkpint，Price-s，Jensen's model，Estimacs)，只有少数几种模型公开发表(如：COCOMO，Softcost，Bailey-Basili's Metamodel)。在公开的模型中，COCOMO被广泛地接受和使用。

Barry Boehm教授在其著作《软件工程经济学》中发表了早期的COCOMO模型(COCOMO 81)，它包括三个层次，即基本COCOMO、中级COCOMO和详细COCOMO模型，其中中级COCOMO在80年代早期被广泛使用。随后，为了支持Ada项目评估，开发了Ada COCOMO模型，对成本驱动因子做了适当调整。1990年后，软件项目管理和开发技术与工具发生了很大的变化，出现了快速应用开发模型、软件重利用、再工程、CASE、面向对象方法以及软件过程成熟度模型等一系列软件工程方法和技术。早期的COCOMO模型已经不再适应新的软件成本估算和过程管理的需要，因此，1994年Boehm重新研究和调整原有模型，根据未来软件市场的发展趋势，发表了COCOMO II。

目前有许多支持COCOMO的商业工具被开发，例如CoCoPro，CB COCOMO，COCOMOID，Costar，COSTMODL，GECOMO Plus，GHL COCOMO SECOMO，CoolSoft，SWAN等等，COCOMO正在成为软件成本估算的标准。

2 COCOMO II的改进

COCOMO模型的演化反映出软件工程技术在过去的20年间不断走向成熟，例如TURN(计算机周转时间)是COCOMO 81的成本驱动因子，当时的程序员用了大量时间等待主机返回批处理任务的结果。今天大多数的程序员使用PC来完成任务，这个参数已经没有意义，在COCOMO II中不再使用。下面总结了COCOMO II的主要变化：

(1)COCOMO II使用三个螺旋式的生命周期模型：applications composition，early design，post-architecture(应用构图、早期设计和后体系结构)。

(2)使用五个规模因子计算项目规模经济性的幂指数B，代替了原来按基本、中期和详细COCOMO模型的不同使用固定指数的方法。

(3)扩展功能点测量，使用源代码行(SLOC)代替DSI。

(4)新增加成本驱动因子：DOCU，RUSE，PVOL，PEXP，LTEX，PCON，SITE。

(5)删除成本驱动因子：VIRT，TURN，VEXP，LEXP，MODP。

(6)改变原有成本驱动因子的参数值，以适应当前的软件测度技术。

收稿日期：2000-04-16

3 模型简要描述

3.1 基于软件市场发展趋势的三个模型

正像COCOMO模型中描述的那样，将来的软件项目必须适合特定的软件过程驱动因素的需要，满足软件重利用、用户需求理解层次、市场和进度限制、可靠性要求。在现代软件工程研究成果的基础上，它将未来(2005年)的软件市场划分为基础软件、系统集成、程序自动化生成、应用集成、最终用户编程五个部分。COCOMO II通过三个生命周期：应用构图、早期设计和后体系结构的螺旋式的模型结构，支持上述五种软件项目。应用构图模型是指通过原型来解决人机交互、系统接口、技术成熟度等具有潜在高风险的内容，通过计算屏幕、报表、第三代语言模块的对象点数来评估软件成本；早期设计模型用于支持确立软件体系结构的生命周期阶段，包括使用功能点和5个成本驱动因子；后体系结构模型是指在项目确定开发之后，对软件功能结构已经有了一个基本了解的基础上，通过源代码行数或功能点数来计算软件工作量和进度，使用5个规模度量因子和17个成本驱动因子调整计算公式。

3.2 工作量及进度计算

COCOMO II仍然使用人月来度量软件开发的工作量。人月是指除去节假日之后一个人在一月内所完成的项目工作量。在COCOMO中，人月与项目进度不同，前者是指工作量，并从中计算开发成本，后者则是指完成项目所需的时间。工作量评估的基本模型如下：

$$PM_{nominal} = A \times (SIZE)^B \tag{1}$$

其中，*SIZE*是估算的软件功能单元的代码行数(以千行为单位)，通过模块功能结构分解和专家法估计，或者使用功能点转化为代码行数。指数*B*反映了项目的规模经济性，当它大于1时所需工作量(*PM*)的增加速度大于软件规模(*SIZE*)的增加速度，体现出规模非经济性；反之，*B*小于1则表示规模经济性。COCOMO使用5个规模度因子W_i，采用公式(2)计算指数*B*。常数*A*通常取值为2.94。

$$B = 0.91 + 0.01\sum W_i \tag{2}$$

5个规模度因子根据其重要性和价值，在6个级别上取值，从非常低到超高，由表1给出(来源于COCOMO II.1999软件包)。

表1 规模度因子

规模度因子W_i	说　明	等　级					
		非常低	低	正常	高	非常高	超高
PREC	有前例	6.20	4.96	3.72	2.48	1.24	0.00
FLEX	开发灵活性	5.07	4.05	3.04	2.03	1.01	0.00
RESL	体系结构和风险控制	7.07	5.65	4.24	2.83	1.41	0.00
TEAM	项目组成员合作程度	5.48	4.38	3.29	2.19	1.10	0.00
PMAT	过程成熟度	7.80	6.24	4.68	3.12	1.56	0.00

要计算项目的进度，使用公式(3)：

$$TDEV = [3.67 \times (PM)^{(0.28+0.2\times(B-0.91))}] \times (SCED\%)/100 \tag{3}$$

其中，*PM*和*B*同公式(1)，*SCED*反映项目组面临的进度压力，是所要求的完成时间与同类项目标准进度之间的比例。

3.3 成本驱动属性

COCOMO模型的最大特色在于其成本驱动因子对工作量的调整，扩展了基本模型。项目双方需要确定各个驱动属性的级别，从公认的值表中得到成本因子的值*EM*。所有成本驱动因子的乘积就是工作量调整值(EAF：Effort Adjustment Factor)。

$$PM_{nominal} = A \times (SIZE)^B \times \prod_{i=1}^{n} EM_i \tag{4}$$

这些精心设计的成本因子一方面体现出产品、硬件、人员、项目因素对软件开发工作量的影响，另一方面也考虑到了可度量性和易用性。根据软件工程理论的发展和外部环境的变化，COCOMO II进一步改进了原有的成本驱动属性，对不同的生命周期阶段模型使用了不同的因子集合。从前面介绍可以知道，各个生命周期阶段模型对项目的理解深度不同，因此能够度量的成本驱动因子也就不同。早期设计模型通常比较简单，作为成本的初步估算，只使用了7个驱动因子(见表2)；后体系结构模型要求对项目的功能和结构详细划分，制定工作任务单元，对各个功能模块有更深刻的认识基础，使用17个详细的驱动因子，分为四个类型：产品属性、计算机平台属性、人员属性、项目属性(见表3)。

表2 早期设计阶段的成本驱动属性

成本驱动因子	说　明	成本驱动因子	说　明
PCPX	可靠性与复杂度	PREX	个人经验
RUSE	可再用性要求	FCIL	开发工具和外部环境
PDIF	计算机开发硬件限制	SCED	进度要求
PERS	个人能力		

表3 后体系结构模型的成本驱动属性

成本驱动因子		说　明	成本驱动因子		说　明
产品属性	RELY	软件可靠性	人员属性	ACAP	分析员能力
	DATA	数据基大小		AEXP	应用经验
	DOCU	文档要求		PCAP	程序员能力
	CPLX	产品复杂度		PEXP	计算机平台经验
	RUSE	可再用性		LTEX	语言和工具经验
				PCON	人员稳定性
计算机属性	TIME	执行时间限制	项目属性	TOOL	使用的软件工具
	STOR	主存限制		SCED	进度要求
	PVOL	平台易变性		SITE	多地点开发

3.4 再利用/再工程模型

为了表示修改现存软件加以再利用对软件成本的影响，COCOMO 81中使用了ESLOC(等价的源指令数目)的概念，通过扩展的线性函数(公式(5)和(6))，计算重利用软件所需要的工作量，其中包括评估更改现有软件以形成新产品的指令数(ASLOC)，以及三个更

改程度参数：DM，设计更改百分比；CM，代码更改百分比；IM，集成百分比。

$$AAF = 0.4 \times DM + 0.3 \times CM + 0.3 \times LM \quad (5)$$

$$ESLOC = ASLOC \times AAF \quad (6)$$

对300多个再利用模块的研究表明，软件再利用的成本函数是非线形的，主要成本由两个方面控制：对要再利用的组件的评估、选择、消化的成本和更改软件的理解成本以及接口测试成本。因此COCOMO II更改了原有模型，增加了更多的调整变量，即评估和选择参数(*AA*)、软件理解参数(*SU*)和程序员的熟悉程度(*UNFM*)。*AA*体现了决定是否软件模块可重利用以及在新产品中集成重利用模块文档的工作量，取值范围为0~8；*SU*则是根据模块的自描述性和耦合程度进行判断，取值为10~50。我们知道一个模块化的、层次清晰的软件能够降低软件的理解成本和相关的接口检查费用，但是程序员对软件的熟悉程度、对软件理解也有很大关系，*UNFM*就是对*SU*的一种补充，其取值范围为0~1。

$$ESLOC = \frac{ASLOC[AA + AAF(1 + 0.02 \times SU \times UNFM)]}{100}, AAF \le 0.5 \quad (7)$$

$$ESLOC = \frac{ASLOC[AA + AAF + SU \times UNFM]}{100}, AAF > 0.5 \quad (8)$$

软件再工程和代码自动转换也是模型额外考虑的地方。使用自动化翻译软件更改的项目通常有更高的代码更改百分比(*CM*)，但是相应的工作量却少得多。COCOMO II使用自动化翻译的代码百分比(*AT*)和劳动生产率(ATPROD)，来计算这种情况对总工作量*PM*的影响。

$$PM_{nominal} = A \times (SIZE)^{B} \times \prod_{i=1}^{17} EM_i + \left[\frac{ASLOC \times \frac{AT}{100}}{ATPROD} \right] \quad (9)$$

4 应用的注意事项

(1)COCOMO模型中规模度因子和工作量调整因子的计算，都是通过以前的项目统计和专家法评估得到的，具有一定的经验性。但COCOMO是一个公开的、灵活的模型，工作量乘数的大小甚至因子本身都不是一成不变，可以根据项目经验作适当地调整。COCOMO II中从三个模型得到评估值E后，使用表4计算相应的输出范围(乐观值、悲观值)。

表4 期望值输出范围

模型	乐观值	悲观值
应用构图	0.50E	2.0E
早期设计	0.67E	1.5E
后体系结构	0.80E	1.25E

(2)早期的COCOMO模型估算应用维护的成本，使用年改变量来估算规模，将成本驱动因子作了相应调整。COCOMO II将重利用模型引入维护阶段，对年改变量<20%的项目不使用维护评估，改用“重利用模型”。对改变量>20%的项目，也不再使用COCOMO81中的可靠性(RELY)和现代编程实践(MODP)因子，而是从重利用模型中取出SU和UNFM作为调整因子。

(3)功能点度量不使用复杂度调整，而是以所采用的编程语言为标准，将未调整的功能点转化为源代码行数。COCOMO II使用表5作这种转换。

表5 功能点与源代码行的转换关系

语言	转换比率 SLOC/FP	语言	转换比率 SLOC/FP
Ada	71	ANSI Cobol 85	91
Ai shell	49	Fortran 77	105
APL	32	Forth	64
Assembly	320	Jovial	105
Assembly(Macro)	213	Lisp	64
ANSI/Quick Basic	64	Modula 2	80
Basic-complied	91	Pascal	91
Basic-Interpreted	128	Prolog	64
C	128	Report Generator	80
C++	29	Spreadsheet	6

5 总结

在过去20年的时间里， COCOMO模型被大量用于软件成本和进度评估。这是一种基于经验的模型，虽然Barry Boehm和他的同事们在COCOMO II中引入了贝叶斯分析，以期提高它的精确度，但仍然不可能做到非常准确。不过COCOMO II确实有了很大提高，它改变了COCOMO 81中软件源代码规模计算困难，一味从经验和类似项目中得到数据，与估算的项目对象联系不大的问题。应用实践表明，今天的COCOMO II模型已经有了相当的正确性，其估算的软件开发成本与实际成本相差不到20%，进度相差不到46%，很好地满足了项目决策和管理的需要。

软件企业能力成熟度的提高不是一蹴而就的，要做到过程持续改进和有效的项目控制，就必须从项目计划开始。 COCOMO 模型给项目管理水平的提高带来了契机，它的输入要求项目管理者细致地划分工作任务结构，输出则给出成本和进度要求，提供给项目双方共同的度量标准。其易用性也广为人们所赞赏。随着软件工程研究实践的深入和理论的发展，COCOMO模型在不断演化。从COCOMO II的变化可以看出，它吸收了对象点、功能点、能力成熟度模型等其它软件工程研究成果，增加了模型的灵活性、可扩展性、可操作性，表现出强大的生命力。

参考文献：

[1] Barry Boehm. 软件工程经济学[M]. 北京：中国铁道出版社, 1990.

[2] Barry Boehm, Bradford Clark, etc. Cost Models for Future Software Life Cycle Process, COCOMO 2.0[M]. USC Center for Software Engineering, 1995.

[3] Barry Boehm. COCOMO II Model Definition Manual[M]. USC Center for Software Engineering.

[4] Roger S Pressman. Software Engineering, A Practitioner's Approach[M]. Fourth Edition, 1997.

[5] Mark C Paulk, etc. The Capability Maturity Model for Software[D]. Software Engineering Institute, Carnegie Melon University.