构造性成本模型 COCOMO*)

杜云梅　李师贤　孙　恒
(中山大学计算机科学系　广州 510275)

摘　要　软件估算已成为众多研究的主题。而构造性成本模型 COCOMO 无疑是其中最为浓墨重彩的一笔。COCOMO 以其独特的优势成为应用范围最广的软件成本估算模型，也是其他众多软件估算方法或模型研究对比、参考和引用的对象。所以对其进行全面而深入的分析是很有意义的。本文在剖析模型原理的同时，把 COCOMO 作为软件成本估算模型的优势、存在的问题及未来的发展之路作为本文讨论的重点。

关键词　构造性成本模型，COCOMO，软件估算

Constructive Cost Model COCOMO

DU Yun-Mei　LI Shi-Xian　SUN Hen
(Department of Computer Science, Sun Yat-sen University, Guangzhou 510275)

Abstract　Software estimation has been the focuse of many research. The constructive cost model, COCOMO, is undoubtedly one of the most significant among them. It is also the studying target of other methods or models. So it is still very meaningful to conduct overall and deep analysis. while analyzing the rationaleof COCOMO models, this paper sets the model advantages, existing problems, future road as the focal point of discussing.

Keywords　Constructive cost model, COCOMO, Software estimation

1　引言

Barry W. Boehm 在 20 世纪 70 年代后期突破性地提出了构造性成本模型(COCOMO)，并在其经典著作文[1]中进行了详细的阐述。凭借其独到的视点、精辟的分析方法和开放的模型细节，该模型立即获得了广大用户的认可，迅速成为世界上应用范围最广的软件成本估算模型。二十多年过去了，COCOMO 依然焕发着璀璨的光芒。为了适应软件现代过程、方法、工具和技术的新发展，Boehm 博士和他南加州大学软件工程中心的同事一起，近几年来不断地对模型进行更新、改进与扩展。通过与软件估算实践者、研究人员、其他估算模型的专家，以及 COCOMO 用户等进行深入的交流与合作，在 90 年代后期逐步形成了 COCOMO II，并在文[2]中给出综合而全面的描述。COCOMO II 是对 1981 年经典 COCOMO 模型的彻底更新，以反映现代软件过程与构造方法。为了避免混淆，就将 1981 年出版的原始 COCOMO 模型称为 COCOMO 81。在 1995 年以前的文献中出现的 COCOMO 一般都是指 COCOMO 81。在 COCOMO II 正式提出之前，还有一个 Ada COCOMO 模型版本。由于其存在的空间比较小，因此本文拟不涉及。COCOMO 81 与 COCOMO II 均指的是理论模型，针对理论模型又有很多的软件实现，即估算工具。通常见到的如"USC COCOMO II. 2000. 0."是指 USC-CSE(南加州大学 软件工程中心)对 COCOMO II 理论模型的软件实现(版本号 2000. 0)。因为 COCOMO 模型内部的基本原理都是开放的，所以目前它已有很多学术的或商业的软件实现。

本文首先对构造性成本模型进行了较为全面的概述，然后在剖析模型原理的同时，把 COCOMO 作为软件成本估算模型的优势、存在的问题及未来的发展之路作为本文讨论的重点。本文的以下部分如此安排：第 2 部分概述 COCOMO 模型的主要框架，并对主要的两个版本进行了对比。第 3 部分分析了模型的基本原理及其适用范围。第 4 部分提出模型目前存在的问题。第 5 和第 6 部分讨论了 COCOMO 的未来发展方向。

2　模型概要

无论是 COCOMO 81 还是 COCOMO II，其基本原理都是：将软件开发所需的工作量表示成软件规模和一系列成本因子的函数。基本工作量估算公式为：

$$PM=A\times S^{E}\times\prod_{i=1}^{n}EM_i \qquad (1)$$

其中，PM 是以人月为单位的软件开发工作量；A 是可校准的常量；S 为软件规模；E 为规模的指数，说明不同规模的软件项目具有的相对规模经济和不经济性；EM 为工作量乘数，反映某个软件项目特征对完成项目开发所需工作量的影响程度；n 为描述软件项目特征的成本驱动因子的个数。

2.1　COCOMO 81

COCOMO 81 模型包括三个详细度和精确度递增的子模型：基本模型、中等模型、详细模型。它用开发模式和成本驱动因子来反映软件项目的特征，开发模式确定工作量估算公式中的系数 A 和指数 E，成本驱动因子的影响表现为工作量乘数 EM，如表 1 所示。

每个成本驱动因子划分为 5 到 6 个级别，每个级别有一个相对应的工作量乘数，反映该成本驱动因子对工作量的影

*)基金项目：广东省自然科学基金项目资助(编号：04009863)。杜云梅　博士生，研究方向为：软件工程。李师贤　教授，博士生导师，研究方向为软件工程，形式语义学等。孙　恒　博士研究生，研究方向为网格计算、语义网格。

响程度。1981 年版本中根据 83 个历史项目数据得出了每个成本驱动因子的每个等级对应的工作量乘数值。

表 1 中等 COCOMO 81 基本估算公式

开发模式	$PM = A \times (KDSI)^E \times \prod_{i=1}^{15} EM_i$ (2)
组织型	A=3.2, E=1.05
半独立型	A=3.0, E=1.12
嵌入型	A=2.8, E=1.20
进度公式：$TDEV = 2.5(MM_{NOM})^{0.35}$	

表 2 COCOMO 81 中等模型的成本驱动因子

产品属性	项目属性
RELY 要求的软件可靠性	MODP 现代编程规范
DATA 数据库规模	TOOL 软件工具的使用
CPLX 产品复杂性	SCED 要求的开发进度
人员属性	计算机属性
ACAP 分析员能力	TIME 执行时间约束
PCAP 程序员能力	STOR 主存储器约束
AEXP 应用经验	VIRT 虚拟机的易变性
VEXP 虚拟机经验	TURN 计算机周转时间
LEXP 编程语言经验	

2.2 COCOMO II

COCOMO II 为不同的软件过程和软件项目开发阶段提供了三个子模型系列：应用组装模型、早期设计模型和后体系结构模型。根据所选择的软件过程、项目目前所处的阶段来决定选用哪个子模型进行估算。COCOMO II 模型以等价源代码行作为软件规模的单位，也可以采用功能点度量方法，模型能自动将未调整功能点估算转换为源代码行来进行估算。特别地，应用组装子模型基于应用点进行估算。早期设计/后体系结构子模型包含 5 个比例因子和 7/17 个成本驱动因子对软件开发工作量的影响。其中比例因子 SF 的综合影响表现为指数 E，成本驱动因子的影响仍然表现为工作量乘数 EM，在同一模型版本中系数 A 设为某一常量，但可以根据实际情况进行校准。

如后体系结构模型的工作量估算公式如下：

$$PM=A\times S^E\times\prod_{i=1}^{17}EM_i$$

其中 $E=B+0.01\times\sum_{j=1}^{5}SF_j$ (3)

开发时间公式为：

$$TDEV=[C\times(PM_{NS})^F]\times\frac{SCED\%}{100} \quad (4)$$

其中 $F=D+0.2\times0.01\times\sum_{j=1}^{5}SF_j$

公式中各符号的含义如表 3 所示。

表 3 公式符号描述

符号	描述
A	可校准的工作量系数，目前设定为 2.94
B	可校准的工作量指数基，目前设定为 0.91
E	工作量比例指数
EM	17 个工作量乘数
SF	5 个比例因子
TDEV	以日历月为单位的开发时间
C	可校准的进度系数，目前设定为 3.67
D	可校准的进度指数基，目前设定为 0.28
F	进度的比例指数
PM_{NS}	不考虑 SCED 成本驱动因子估算人月数

COCOMO 2000 年版本根据精心挑选的 161 个历史项目数据得出了每个成本驱动因子等级对应的工作量乘数值。对于早期设计子模型与后体系结构子模型，规模计算和比例因子都是相同的，而成本驱动因子有很大不同，早期设计模型将后体系结构模型的 17 个因子合并为 7 个较为粗略的因子。表 4，5，6 分别给出了 COCOMO II 的比例因子和成本驱动因子。

表 4 COCOMO II 比例因子

PREC 先例性	TEAM 团队凝聚力
FLEX 开发灵活性	PMAT 过程成熟度
RESL 体系结构与风险化解	

表 5 COCOMO II 后体系结构模型成本驱动因子

产品因子	人员因子
RELY, DATA, CPLX, RUSE(可复用开发), DOCU(匹配生命周期需求的文档)	ACAP, PCAP, APEX, PCON(人员连续性), PLEX(平台经验), LTEX (语言和工具经验)
项目因子	平台因子
TOOL, SCED, SITE(多点开发)	TIME, STOR, PVOL

表 6 COCOMO II 早期设计模型成本驱动因子

RCPX 产品可靠性与复杂性	RUSE 可复用开发
PDIF 平台难度	
PERS 人员能力	PREX 人员经验
FCIL 项目设施	SCED 要求的开发进度

2.3 COCOMO 模型比较

表 7 总结了 COCOMO II 对 COCOMO 81 的主要更新，更详细的模型信息与细节参见文[1,2]和其它参考文献。

表 7 COCOMO II 对 COCOMO 81 的更新与升级

	COCOMO 81	COCOMO II 后体系结构	COCOMO II 早期设计
规模计算	源代码行 (SLOC)	未调整功能点(UFP)或 SLOC	UFP 或 SLOC
复用模型	等价 SLOC=线性 f(DM, CM, IM)	等价 SLOC=非线性 f(AA, SU, UNFM, DM, CM, IM)	等价 SLOC=非线性 f(DM, CM, IM)
规模的指数 E	到决于开发模式	取决于五个比例因子的等级	取决于五个比例因子的等级
成本驱动因子	去掉了 TURN、MODP，增加了 DOCU、RUSE、PCON、SITE；修改了 CPLX、TOOL 的等级定义；更新了对应的全部工作量乘数。		
产品	RELY、DATA、CPLX	RELY、DATA、CPLX、DOCU、RUSE	RCPX、RUSE
平台	TIME、STOR、VIRT、TURN	TIME、STOR、PVOL (=VIRT)	PDIF
人员	ACAP、AEXP、PCAP、VEXP、LEXP	ACAP、PCAP、PCON、APEX、LTEX、PLEX	PERS、PREX
项目	MODP、TOOL、SCED	TOOL、SCED、SITE	SCED、FCIL
需求变更	需求易变性等级 RVOL	需求演进 REVL	REVL
过程模型支持	瀑布模型	瀑布模型、螺旋模型、RUP 等	

3 模型分析与模型应用范围

本质上，COCOMO 模型包括两个底层信息模型。第一个是用于描述软件项目框架，包括过程模型、文化、干系人、方法、工具、开发团队以及软件产品的规模或复杂性等，用规模、比例因子、成本驱动因子及相关定义、公式与表格等来描述。

第二个就是经验库，即精心挑选的历史项目数据。模型根据这些历史数据得出基本的参数值并进行校准，估算者按照要估算项目的具体情况选择最适合的参数值，运用模型公式进而得出估算，即从历史先例估算出未来项目可能需要的相关资源(工作量与时间)。软件项目的成本依赖于项目的状态和特征，应用该模型进行软件估算的过程也是把握并预测软件项目特征的过程，借助 COCOMO 的项目框架，可以更全面更系统地利用项目的可用信息，预测有价值的未知信息；我们还可以针对要估算项目的特征，选择最恰当的经验库来确定模型参数并进行校准，从而使得出的估算更可信、更准确。较为详细的描述参见参考文[1,2]。

从另一方面看，估算公式中的 A 可以看作是代表生产率的常量，反映模型用户机构的平均软件开发生产率水平；ΠEM_i 可以理解为相对于平均生产率常量，具体软件项目的特殊属性导致的生产率的倍数差异；指数 E 反映项目在具体环境条件下将显示出的规模经济或不经济性。从而我们可以从生产率的角度分析项目成本与进度，参考文[3]给出了重要启示。也可以利用该模型开展机构软件开发生产率的研究，发现关键薄弱环节进而找到提高生产率的途径。另外根据规模经济或不经济性分析，模型可以帮助我们做出某些决策，如过程改进决策、分析机构从 SEI-CMM 2 级提升到 SEI-CMM 3 级的投资是否值得。

COCOMO 模型包含大量的成本驱动因子，它们基本上抓住了影响开发工作量的各个方面的软件项目特征，我们也可以据此进行某个或某些因子的敏感性分析，进而识别项目的高风险因素，以帮助进行风险管理。文[4]就是一个很好的例子。

一般说来，COCOMO 是一个计划和执行软件项目的目标成本模型，它是管理软件项目或商业软件生产线的重要组成部分。成本模型提供了软件开发干系人之间交流商务决策的框架。COCOMO II 支持合同协商、过程改进分析、工具购买、体系结构变更、组件开发或购买权衡、成本与风险管理、开发与复用决策、遗留软件淘汰决策，以及其它一些有可信估算基础的投资收益决策。这正是 COCOMO 的优势所在，模型不仅给出有参考意义的资源估算结果，而且更重要的是能让用户从经济学的角度看待整个项目，可视地分析多种方案、多种情况下的成本与进度，从而做出让干系人都认可、或最具成本效益的软件决策。

4 模型的优点与存在的问题

该模型得到广泛的应用与认可，自然具有其他方法与模型无法比拟的优点。

首先，COCOMO 模型是开放的、面向公众领域的。除了历史经验库由于保密协议而没有公开以外，其它所有定义、公式、原理、方法、参数值、假设条件等都是对公众开放的。我们可以从其网站、用户手册、著作等形式的资料中得到有关模型的所有细节。

其次，COCOMO 模型是完全文档化的。模型的开创者与研究者本着科学严谨的态度，将研究方法、模型示例、数据收集表单、分析与校准过程等都形成文档形式。这不仅有助于模型的后续研究，也能帮助用户更彻底地理解模型。

第三，COCOMO 模型是数据驱动的。COCOMO II 是基于 161 个历史项目数据，分别来自商业、航空、政府和一些非盈利机构。这些数据都是经过严格筛选，完全符合 COCOMO 模型对相关数量的定义。估算本来就是根据过去预测未来的过程，只有以可靠准确的历史数据作为依托，得出的估算才具有说服力。这也带来另外一个优点，模型使用者可以用更符合本机构甚至本项目的历史数据集，重新校准模型的参数与定义，从而得到更适合本机构的模型版本。

另外，COCOMO 模型本身的研究过程，也为软件行业的研发项目树立了良好的榜样：开放、合作、沉稳。为了跟上软件工程发展的步伐，提出了彻底的更新与扩展；为了弥补大学科研环境与实践的脱节，他们用会员制度与广大用户和软件机构进行深入的合作；从开始着手更新 COCOMO 81 到 COCOMO II 的基本形成，耗时四年多时间，与盛行的浮躁之风和急于进入市场思想形成极大反差。从开放而全面的文档中我们能深切地体会到这一点。

当然，软件估算本身就是非常复杂和困难的。COCOMO 作为其中的一种成本估算方法，不可避免地有其问题与不足。

它只估算软件开发的工作量，不包括需求阶段、交付与运行阶段的工作量。COCOMO 关注焦点是瀑布生命周期模型中的产品设计、编码、集成与测试，在 RUP(Rational Unified Process)生命周期模型中的细化与构造阶段，即它不包括瀑布模型中的计划与需求阶段、运行与维护阶段，RUP 模型中的起始阶段、交付阶段的工作量。在使用该模型进行估算时，一定要注意到这一点，并对估算结果进行调整，因为需求可能花费大量的工作量。

COCOMO 起初是为大型项目而构想的估算模型，对于低于两人年工作量的项目估算，通常是不合理的。对于小型项目而言，估算过程过于繁琐，涉及太多的细节，有点杀鸡用牛刀的味道。较为简单的应用组装与早期设计子模型，也不能适用于很多类型的中小型项目。

另外一点就是估算不准的问题。有部分模型使用者抱怨说费了半天劲估算的结果却与实际情况相去甚远。应该辩证地来看待这个问题。不只是 COCOMO，其它所有的软件估算模型或多或少都存在不够准确的问题[5~7]。原因首先是估算本身的困难性、预测固有的不确定性，能保证完全准确那就不叫估算而应该称为计算了。另外一个造成结果误差太大的原因是模型输入信息的不准确性，对模型变量与范围定义缺乏理解甚至误解、数据收集中的不一致、估算者对项目情况的主观偏见、不恰当的假设，都可能带来很大差异。从这一方面可以说该模型是个“垃圾入-垃圾出”的模型，如果希望得出的估算结果更为准确，就应该尽可能多地得到尽可能准确的项目信息。

其实 COCOMO 模型的精髓就在于：估算者能完全理解模型的输入、输出、内部模型和假设条件，针对具体情况进行权衡分析、敏感性分析等，对其结果也能有十分合理的解释。如果不加理解地套用公式与参数，估算结果的可信度就可想而知了。估算者自己应该先端正一个思想，软件估算本来就是一个根据已知预测未知的过程。对可知的信息把握得越充分、了解得越透彻，对未知的估算才有可能更准确。那种寄希望于估算工具的神力，想简单地输入两个数据，就得出完全准确估算的幻想终究是不切实际的。保持一种开明沉稳的态度，才能发挥 COCOMO 模型巨大的参考价值。

5 未来之路

目前，为了弥补 COCOMO II 中存在的问题与不足，已经提出了一些扩展模型，如：

COPSEMO (COnstructive Phased Schedule and Effort MOdel):构造性阶段进度和工作量模型,将 COCOMO II 所计算的工作量和进度,分布或映射到 MBASE/RUP(Model-Based[System] Architecting and Software Engineering and Rational Unified Process)的阶段上。

CORADMO (COnstructive Rapid Application Development MOdel):快速应用开发(RAD)策略当前已获得普遍的接受,COCOMO II 却没有提到任何用于减少进度的快速应用开发策略,所以专门以快速应用开发估算模型 CORADMO 来加以弥补。

COCOTS(COnstructive COTS integration cost model):用 COTS 组件构造系统将变得越来越重要,而它与传统意义上的开发与复用有很大差异,它在带来便利的同时也带来了一大堆风险。这是 COCOMO II 没有建模的情况,从而专门设计了 COTS 集成成本模型 COCOTS 来反映。

COQUALMO(COnstructive QUALity MOdel):成本、进度与质量是软件开发中高度相关的三个因素。基于缺陷引入和缺陷消除子模型,形成了对 COCOMO Ⅱ 的质量模型扩展 COQUALMO。

COPROMO(COnstructive PROcess-improvement MOdel):构造性过程改进模型(COPROMO)关注于估算投入资源(如新技术或过程)分配的成本效益,从而改进生产率,是一个用于软件工程高级管理部门的策略性计划决策辅助模型。

专家 COCOMO:通过识别、分类、量化和按主次排列项目风险,来帮助项目计划。除了常规的成本和进度计算,它还检测成本估算输入异常并提供风险控制建议。

这些模型扩展很好地弥补了 COCOMO II 的不足,但研究才刚刚起步。它们将使 COCOMO II 更加完善,但显然也将使之更为庞大和复杂。如何有机地组织好这些扩展,也是个值得注意的问题。

我们不能要求每个模型使用者都成为估算专家,精通 COCOMO II 的所有定义、假设与原理。尽可能多地了解模型细节和可知的项目信息,这是用户一方为获得更准确的估算而做出的努力。同时,模型的研究者也应该尽量使模型向易用、精简的方向发展。如果需要很长的学习周期才能着手使用,无疑给模型设下一道继续发展的屏障。

与经济学更紧密的结合,或许是 COCOMO 在越来越丰富的软件估算模型中突出的途径之一。成本效益、权衡、风险、敏感性分析,以及其它支持软件决策的分析,似乎越来越成为该模型的闪光之处。

结论与展望 构造性成本模型以其开放的模型细节、突破性的估算方法、历史数据的说服力、有理有据的分析过程等,当之无愧地成为应用范围最广的软件成本估算模型。说起软件成本估算,无不提及 COCOMO。它已成为软件估算领域的必修模型,也是其他众多软件估算方法与模型对比、参考和引用的对象,以之为中心也展开了很多的应用研究。

由于估算本身的困难与复杂性,COCOMO II 模型无法回避的局限性,它也受到了来自应用、研究与理论领域的种种质疑。这是模型发展的动力,同时模型研究和使用者必须清楚该模型的应用范围与局限性,才能扬长避短,充分发挥其价值。

国内目前对软件估算的呼声逐渐多起来,有关软件估算的疑问也随处可见。软件项目管理、软件工程方面的书籍或培训课程都会提及软件成本估算。COCOMO 是必须讲述的模型之一。但总的说来还处于简单套用缺乏理解的阶段。再加上国内外软件发展阶段的差距、软件机构文化上的不同、而 COCOMO 又是以国外历史项目数据为依托,所以估算结果的准确性非常令人怀疑。根据本地经验数据和过程模型进行校准与裁剪,应该是我们成功应用 COCOMO II 模型的第一个步骤。

参 考 文 献

1 Boehm B W. Software Engineering Economics. Prentice Hall PTR, 1981

2 Boehm B W, Abts C, Brown A W, Chulani S, Clark B K, Horowitz E, Madachy R, Reifer D, Steece B. Software Cost Estimation with COCOMO II. Prentice Hall PTR, 2000

3 Boehm B. Safe and Simple Software Cost Analysis. Software, IEEE, 2000,17(5)

4 Musilek P, Pedryez W, NanSun, Succi G. On the Sensitivity of COCOMO II Software Cost Estimation Model. 002 IEEE

5 Briand L C, Elmam K, Surmann D, Wieczorek I, Maxwell K D. An Assessment and Comparison of Common Software Cost Estimation Modeling Techniques. Software Engineering, 1999

6 Kavoussanakis K, Sloan T. UKHEC Report on Software Estimation, 2002

7 Chatzoglou P D, Macaulay L A. A review of existing models for project planning and estimation and the need for a new approach. Project Management, 1996, 14(3)

8 Tian Liang, Noore A. Multistage software estimation. System Theory, 2003. In: Proc. of the 35th Southeastern Symposium on, 2003

9 Clark B K. Cost Modeling Process Maturity-COCOMO 2.0. Aerospace Applications Conference. In: Proc. IEEE, Feb. 1996,3

10 Wang Shouli, Kountanis D. IASCE-an intelligent assistant to software cost estimation. In: Proc. of the 1992 IEEE Int. Conf. on Tools with AI Arlington, VA, Nov. 1992

11 Tharp T, Zalewski J. Economics and Software Engineering: Transdisciplinary Issues in Research and Education, 2002

12 Jones C. Patterns of Software Systems Failure and Success. International Thomson Press, 1996

13 JØrgensen M. Top-down and bottom-up expert estimation of software development effort. Information and Software Technology, 2004, 46

14 McConnell S. Rapid development: taming wild software schedules. Microsoft Press, 1996

15 Jorgensen M, Sjoberg D I K. Impact of effort estimates on software project work. 2001