COCOMO-II模型 (1)

- COCOMO 81模型适用于专用的定制的软件项目，它建立在“瀑布模型”的过程框架上
- 1997年 Boehm 等人提出来的 COCOMO-II 模型则适用于广泛汇集各种技术的软件项目，如商用软件、面向对象软件、通过螺旋型或演化型开发模型制作的软件

1

1

COCOMO-II模型(2)

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

2

2

COCOMO-II模型(3)

- COCOMO II 模型通过三个不同的模型分别对三个不同的阶段进行估算:
  - 应用组装模型（Applications Composition）
  - 早期设计模型（Early Design）
  - 后架构模型（Post-Architecture）

3

3

COCOMO-II模型(4)

应用组装模型

- 用于早期原型开发的工作量估算
- 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
- 模型使用“对象点”，而不用“源代码行”或“功能点”进行估算

4

4

COCOMO-II模型(5)

- 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
- 后架构模型
  - 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

5

5

COCOMO-II模型(6)

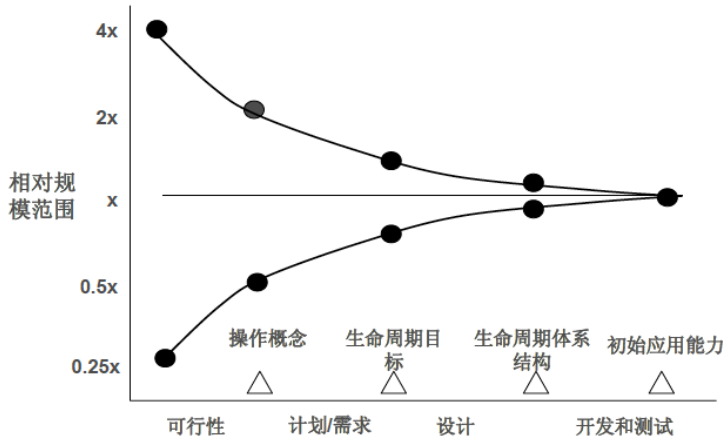
- 模型可裁剪组合原理所基于的假设：
  - 现在和未来的软件项目将按特有的过程驱动因子来裁剪其过程，而不是单一的瀑布模型；
  - 所用软件成本估算模型的粒度需求与支持软件成本估算的可用信息的粒度一致；
  - COCOMO II能在项目早期提供粗粒度的成本驱动因子信息，在后期逐步提供更详细的信息，产生依赖于估算输入定义程度的范围估算

6

6

COCOMO-II模型(7)

- 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

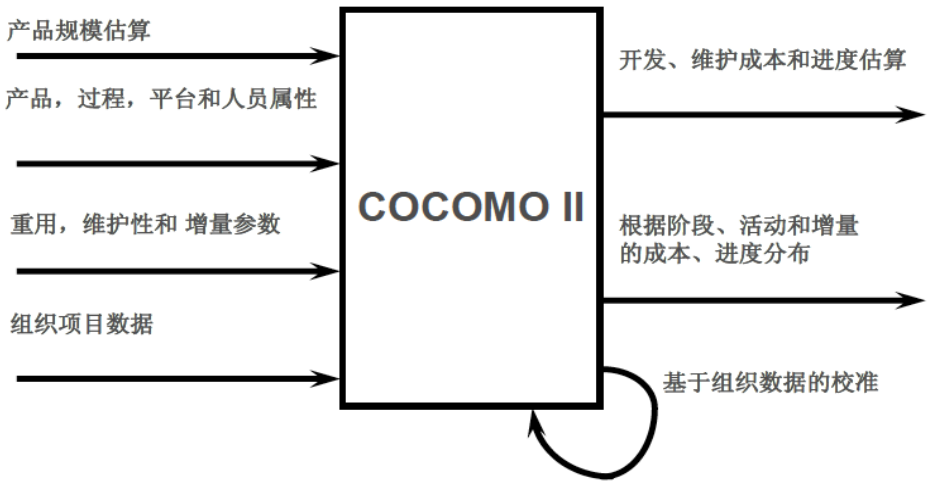


7

7

COCOMO-II模型(8)

COCOMO 黑盒模型



8

8

5.1 应用组装模型
对象点/应用点

- 对象点是1991年由 Banker、Kauffman和Kumar等人提出的
- 它类似于功能点，是一种软件间接度量
- 根据（用户界面）屏幕（screen）数、报告（report）数、建造应用所需使用的第三代语言（3GL）构件数来计数。屏幕、报告和3GL构件统称为元素。
- 2000年Boehm等人在COCOMO II: 2000中把“对象点”改为“应用点”，以避免概念的混淆

9

9

应用组装模型估算的步骤

1. 评估应用计数：估算组成该应用的屏幕、报告和3GL构件数目
2. 确定复杂性级别：对于每一个屏幕、报告、3GL构件，根据一些特征，把它们划分到简单、中等和困难等3个复杂性级别

10

10

表1 屏幕应用点复杂性等级

包含的视图数	数据表的数量 来源		
	总数 < 4 (< 2个srvr, < 3个clnt)	总数 < 8 (< 2~3个srvr, < 3~5个clnt)	总数 ≥ 8 (> 3个srvr, > 5个clnt)
< 3	简单的	简单的	中等的
3 ~ 7	简单的	中等的	困难的
≥ 8	中等的	困难的	困难的

11

11

表2 报告应用点复杂性等级

包含的节数	数据表的数量和来源		
	总数 < 4 (< 2个srvr, < 3个clnt)	总数 < 8 (2~3个srvr, < 3~5个clnt)	总数 ≥ 8 (> 3个srvr, > 5个clnt)
< 2	简单的	简单的	中等的
2 ~ 3	简单的	中等的	困难的
≥ 4	中等的	困难的	困难的

- 每一个报告可以包括有若干节（sections）
- 根据报告所涉及节数和数据表数，可以确定该报告的复杂性等级。

12

12

应用组装模型估算的步骤

1. 评估应用计数：估算组成该应用的屏幕、报告和3GL构件数目
2. 确定复杂性级别：对于每一个屏幕、报告、3GL构件，根据一些特征，把它们划分到简单、中等和困难等3个复杂性级别

应用组装模型估算的步骤

3. 加权：根据每一个元素的复杂性级别，参照表3对其加权

表3 应用点复杂性加权

元素类型	复杂性加权		
	简单的	中等的	困难的
屏幕	1	2	3
报告	2	5	8
3GL构件	-	-	10

应用组装模型估算的步骤

4. 计算应用点数：将每一个元素的计数乘以权值得到该元素的加权计数，再将各个元素的加权计数累加，得到总的应用点计数

5. 估计项目复用的百分比r：如果项目在开发中使用了构件或复用了以前的软件，再估计复用的百分比r

15

15

应用组装模型估算的步骤

6. 计算要开发的新应用点数：通过以下公式得到调整后的新应用点数NAP

NAP =（应用点计数）×（100 – r）/ 100

例如，一个应用程序包含 840 个应用点，其中20%可以通过使用现成的构件来提供，那么调整后的新应用点（NAP）的得分将是

NAP = 840×(100 - 20) / 100 = 672

16

16

应用组装模型估算的步骤

7. 确定生产率：其单位是PROD，即 NAP / 人月

开发者的经验和能力 / 开发环境成熟度和能力	很低	低	标称	高	很高
生产率PROD	4	7	13	25	50

17

17

应用组装模型估算的步骤

8. 估算工作量

PM = NAP / PROD

- 例如，一个应用程序有 672 个新应用点，开发环境的生产率是正常的，则项目的估计工作量为：

PM = 672 / 13 = 52 （人月）

18

18

(2) 早期设计模型

1. 估算功能点: 根据软件需求和设计文档的信息，对于每一个功能类型，分别统计功能计数
2. 确定复杂性等级: 按照下表，确定每个功能的复

对于内部逻辑文件和外部接口文件			
记录元素类型数			
	1～19	20～50	≥51
1	低	低	一般
2～5	低	一般	高
≥6	一般	高	高

19

19

FP复杂性等级表

对于外部输出和外部查询			
文件类型数	数据元素类型数		
	1～5	6～19	≥20
0～1	低	低	一般
2～3	低	一般	高
≥4	一般	高	高

对于外部输入			
文件类型数	数据元素类型数		
	1～5	6～19	≥20
0～1	低	低	一般
2～3	低	一般	高
≥4	一般	高	高

20

应用早期设计模型的步骤

3. 对各功能类型计数加权：根据下表，按照复杂性等级对各个功能类型计数加权（该权值反映了实现功能所需工作量的大致估计）

功能类型	复杂性权值		
	低	一般	高
外部输入	3	4	6
外部输出	4	5	7
外部查询	3	4	6
内部逻辑文件	7	10	15
外部接口文件	5	7	10

21

应用早期设计模型的步骤

4. 计算未调整的功能点：把所有加权后的功能计数相加，得到未调整功能点
5. 把未调整功能点转换成源代码行数

- 传统的功能点度量，还需要考虑 14 种用于校正度量值的影响因素，然而COCOMO II 没有这样做。COCOMO II 先计算未调整功能点，再应用复用因子、成本驱动因子、尺度因子进行调整。

22

应用早期设计模型的步骤

- 例如，项目的功能点数据如下：

功能类型	类型	数据元素	复杂性	权重	功能计数
外部输入	文件1	8	低	3	4
外部输出	文件1	12	低	4	7
外部查询	文件7	16	高	6	5
内部逻辑文件	记录22	104	高	15	7
外部接口文件	记录14	56	高	10	6

- 由此可计算未调整功能点为
  $UFP = 4 \times 3 + 7 \times 4 + 5 \times 6 + 7 \times 15 + 6 \times 10$
  $= 235$

23

从UFP到SLOC的默认转换率

语言	SLOC / UFP	语言	SLOC / UFP
机器代码	640	Lisp	64
基本汇编	320	Prolog	64
宏汇编	213	C++	55
C	128	Java	53
Fortran 77	107	Ada 95	49
Unix Shell Scripts	107	AI Shell	49
Compiled Basic	91	Visual C++	34
Pascal	91	Visual Basic 5.0	29
Cobol (ANSI 85)	91	PERL	27
Modula 2	80	SQL	20
Fortran 95	95	PowerBuilder	16
Forth	64	HTML 3.0	15

24

工作量估算方法

- 工作量估算公式为：

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

- 其中，$A = 2.94$（对COCOMO II：2000）
- $KSLOC$ 是千源代码行数
- 指数 E 由5 个尺度因子计算得到

$$E = B + 0.01 \times \sum_{i=1}^{5} SF_i$$

- 其中，$B = 0.91$（对COCOMO II: 2000）
- EM_i 是工作量调整因素中的成本驱动因子

25

尺度因子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，权重的值称为尺度因子（SF）
- 五个尺度因子：
  - 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)

26

尺度因子SF_i的含义

尺度因子	解　释
先例性 SF_1	反映机构对此类型项目的经验。包括机构对产品目标的理解程度，以往类似系统的工作经验，相关硬件和操作程序开发的熟悉和协同程度，数据处理、体系结构以及算法是否需要有创新等。“很低”代表无经验，“很高”代表对该领域有彻底了解。
开发灵活性 SF_2	反映开发过程中的灵活程度。“很低”代表软件范围必须与已建立的需求、外部接口规范等高度一致，“很高”代表客户只给出了总的目标。
体系结构/风险化解 SF_3	反映风险分析情况。包括是否识别出关键风险项，软件体系结构在任务、接口、构件、技术、性能方面不确定性如何，是否通过设计评审和完善体系结构建立了化解这些风险项的里程碑。“很低”代表无分析，“很高”代表完全、彻底的风险分析。

27

尺度因子SF_i的含义–续

尺度因子	解　释
团队凝聚力 SF_4	反映开发团队相互了解和协作的程度。包括项目相关人员的目标和企业文化的一致性，项目相关人员是否有能力有意愿适应其他相关人员的目标，团队建设和团队工作是否有经验。“很低”代表交互很困难，“很高”代表团结高效。
过程成熟度 SF_5	反映机构的过程成熟度。可用 5 减去CMM过程成熟度等级得到。

- 通过分析上表所示的 5 个尺度因子来计算指数E。这些因子有六个等级，从“很低”到“极高”，分别赋予 5～0 值，将这些估算值相加除以 100，再加上 $B=0.91$，就得到该指数的取值。

28

尺度因子SF_i的含义-- 实例

- 一个组织正承担一个项目，该组织对于该项目所在领域没有经验。项目客户没有定义需采用的过程，需求和接口只有大概的构想。在项目进展中没有做重大风险分析，还需组织新的开发团队来完成这个系统。此外该组织最近刚实行过程改善计划，并且依据CMM模型被评为2级。

29

尺度因子SF_i的含义-- 实例

- 在进行指数计算时，各尺度因子取值为：
  (1) 先例性：机构的新项目，取值“低”(4)
  (2) 开发灵活性：无客户介入，取值“很高”(1)
  (3) 体系结构/风险化解：无风险分析，取值“很低”(5)
  (4) 团队凝聚力：新团队，取值“一般”(3)
  (5) 过程成熟度：有些过程控制，取值“一般”(3)
- 计算得到的指数 E 为：

$$E = B + 0.01 \times \sum_{i=1}^{5} SF_i$$
$$= 0.91 + 0.01 \times (4+1+5+3+3) = 1.07$$

30

成本驱动因子(1)

- 后架构模型采用 17 个成本驱动因子 EM_i，来调整标称工作量，以反映待开发软件的特征
- 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

31

成本驱动因子(2)

- 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

32

成本驱动因子及工作量系数表

- 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状况

EM_i	描述	很低	低	标称	高	很高	极高
产品							
RELY	软件可靠性	0.82	0.92	1.00	1.10	1.26	—
DATA	数据库规模	—	0.90	1.00	1.14	1.28	—
CPLX	产品复杂性	0.73	0.87	1.00	1.17	1.34	1.74
RUSE	可复用开发	—	0.95	1.00	1.07	1.15	1.24
DOCU	文档编制	0.81	0.91	1.00	1.11	1.23	—

33

成本驱动因子及工作量系数表(续)

EM_i	描述	很低	低	正常	高	很高	极高
平台							
TIME	执行时间限制	—	—	1.00	1.11	1.29	1.63
STOR	内存限制	—	—	1.00	1.05	1.17	1.46
PVOL	平台易变性	—	0.87	1.00	1.15	1.30	—
人员							
ACAP	分析员能力	1.42	1.19	1.00	0.85	0. 71	—
PCAP	程序员能力	1.34	1.15	1.00	0.88	0.76	—
PCON	人员连续性	1.29	1.12	1.00	0.90	0.81	s

34

成本驱动因子及工作量系数表(续)

EM_i	描述	很低	低	正常	高	很高	极高
APEX	应用经验	1.22	1.10	1.00	0.88	0.81	—
PLEX	平台经验	1.19	1.09	1.00	0.91	0.85	—
LTEX	语言和工具经验	1.20	1.09	1.00	0.91	0.84	—
项目							
TOOL	软件工具	1.17	1.09	1.00	0.90	0.78	—
SITE	多站点开发	1.22	1.09	1.00	0.93	0.86	0.80
SCED	开发进度	1.43	1.14	1.00	1.00	1.00	—

35

各成本驱动因子等级的划分

1）要求的可靠性：主要看软件失效造成的影响

很低	低	标称	高	很高	极高
仅有点不方便	低度易弥补的损失	中度易弥补的损失	高财务损失	性命攸关	n/a

2）数据库规模：因为测试数据库需大量测试数据，可用 D/P 即测试数据的字节数与程序SLOC的比率来衡量产生和维护这些测试数据所需工作量

很低	低	标称	高	很高	极高
n/a	$D/P<10$	$10\leqslant D/P<100$	$100\leqslant D/P<1000$	$1000\leqslant D/P$	n/a

36

各成本驱动因子等级的划分

3）产品复杂性：从5个方面来综合衡量产品复杂性

	很低	低	标称	高	很高	极高
控制操作	控制转移少，简单模块连接	结构化编程，多为简单谓词	简单回调和消息传递，分布式处理	存在复合谓词和高度嵌套	多任务复杂分布式处理	多资源调度，微指令控制，分布式硬实时控制
计算操作	简单表达式计算	中等表达式计算	标准算法和矩阵运算	基本数值分析算法	复杂数值分析算法	随机数据分析，并行计算
设备相关操作	有简单格式的简单读写语句	在GET / PUT级读写	包括状态检查、错误处理的读写	物理 I/O 操作，优化I/O	通信链路处理、嵌入式系统	微程序控制操作、性能关键嵌入式系统
数据管理操作	主存的简单数组，简单查询	单文件子集、中等查询	多文件读写和复杂数据查询、更新	数据流触发器，数据重构	分布式数据库、复杂触发器	高耦合动态相关的对象结构，自然语言数据管理
用户界面操作	简单输入表单，报表生成器	简单GUI生成器	工具集的简单使用	工具集开发扩展，多媒体	中等2/3D动态图形和多媒体	复杂多媒体、虚拟现实、自然语言界面

37

各成本驱动因子等级的划分

4）可复用开发：主要考虑构造在当前或未来项目中复用的构件所需要的额外工作量

很低	低	标称	高	很高	极高
n/a	无	跨项目	跨程序	跨产品线	跨多条产品线

5）匹配生命周期需求的文档编制：主要考虑在软件产品生命周期各阶段对项目文档的需求的适应性来评估

很低	低	标称	高	很高	极高
未满足绝大多数生命周期需求	未满足某些生命周期需求	满足生命周期需求	超出生命周期需求	大大超出生命周期需求	n/a

38

各成本驱动因子等级的划分

6) 执行时间限制：主要通过系统或子系统预期消耗的时间与可用执行时间的比率（%）来确定其执行时间的限制等级：

很低	低	标称	高	很高	极高
n/a	n/a	<50%	50～70%	70～85%	85～95%

7) 内存限制：主要通过系统或子系统预期使用的内存空间与可用空间的比率（%）来确定其内存限制的等级：

很低	低	标称	高	很高	极高
n/a	n/a	<50%	50～70%	70～85%	85～95%

39

各成本驱动因子等级的划分

8) 平台易变性：平台是指硬件和支持软件的总体（旧称虚拟机）

很低	低	标称	高	很高	极高
n/a	每1年有主要变更，每1个月有次要变更	每6个月有主要变更，每2周有次要变更	每2个月有主要变更，每周有次要变更	每2周有主要变更，每2天有次要变更	n/a

9) 分析员能力：主要考虑分析和设计人员的分析和设计能力、生产率和彻底性、沟通和协作能力：

很低	低	标称	高	很高	极高
15分	35分	55分	75分	90分	n/a

40

各成本驱动因子等级的划分

10) 程序员能力：能力评价应基于程序员作为小组的能力而不是作为个人能力。主要考虑能力、生产率、彻底性、沟通和协作能力，不考虑程序员的经验。

很低	低	标称	高	很高	极高
15分	35分	55分	75分	90分	n/a

11) 人员连续性：主要根据人员的年周转率来确定：

很低	低	标称	高	很高	极高
48% / 年	24% / 年	12% / 年	6% / 年	3% / 年	n/a

41

各成本驱动因子等级的划分

12) 应用经验：项目组对应用领域的经验

很低	低	标称	高	很高	极高
≤2个月	6个月	1年	3年	6年	n/a

13) 平台经验：平台使用经验对生产率也有影响。平台包括GUI、数据库、网络和分布式中间件等。

很低	低	标称	高	很高	极高
≤2个月	6个月	1年	3年	6年	n/a

14）语言和工具经验：除了编程语言方面的经验外，项目支持工具集方面的经验也影响开发工作量。

很低	低	标称	高	很高	极高
≤2个月	6个月	1年	3年	6年	n/a

42

各成本驱动因子等级的划分

15)软件工具的使用：

很低	低	标称	高	很高	极高
编辑编码调试工具	简单的前端/后端CASE，很少集成	基本生命周期工具，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	n/a

16)多点开发：主要根据两个因素来划分等级。

	很低	低	标称	高	很高	极高
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
沟通描述	电话邮件	专用电话、传真	宽带电子邮件	窄带电子邮件	窄带电子邮件、视频会议	多媒体交互

43

早期设计模型的工作量调整

- 采用了后架构模型成本驱动因子的简化集。

因素	描 述	结合后架构成本驱动因素
RCPX	产品可靠性和复杂性	RELY, DATA, CPLX, DOCU
RUSE	必要的复用	RUSE
PDIF	平台困难程度	TIME, STOR, PVOL
PERS	人员能力	ACAP, PCAP, PCON
PREX	人员经验	AEXP, PEXP, LTEX
FCIL	设施	TOOL, SITE
SCED	进度	SCED

44

早期设计模型的工作量调整

产品可靠性和复杂性（RCPX）							
强调可靠性和文档化	很少	少	有些	基本	强烈	很强烈	极强烈
产品复杂性	很简单	简单	有些	中等	复杂	很复杂	极复杂
数据库规模	小	小	小	中等	大	很大	极大
等级	极低	很低	低	标称	高	很高	极高
EM_i	0.49	0.60	0.83	1.00	1.33	1.91	2.72

平台困难程度（PDIF）					
时间和存储限制	≤50%	≤50%	65%	80%	90%
平台易变性	很稳定	稳定	有些不稳定	不稳定	极不稳定
等级	低	标称	高	很高	极高
EM_i	0.87	1.00	1.29	1.81	2.61

45

早期设计模型的工作量调整

人员能力（PERS）							
分析员和程序员能力	20分	35分	45分	55分	65分	75分	85分
人员连续性	45%	30%	20%	12%	9%	6%	4%
等级	极低	很低	低	标称	高	很高	极高
EM_i	2.12	1.62	1.26	1.00	0.83	0.63	0.50

人员经验（PREX）							
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
EM_i	1.59	1.33	1.22	1.00	0.87	0.74	0.62

46

早期设计模型的工作量调整

TOOL支持	少	一些	简单的CASE工具集	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
多点开发	对多点复杂开发（M/S）有少量支持	对复杂的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开发有强力支持	对简单的M/S开发有强力支持	对同处一地或简单的M/S开发有超强支持
等级	极低	很低	低	标称	高	很高	极高
EM_i	1.43	1.30	1.10	1.00	0.87	0.73	0.62

- 可复用开发（RUSE）与开发进度限制（SCED）与后架构模型的等级划分一致。

47