第16卷 第12期
2006年12月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol.16 No.12
Dec. 2006

扩展的用例点估算方法

周 杨,吴海涛,张栋伟
(上海师范大学 数理信息学院,上海 200234)

摘 要:通过分析用例点估算技术的不足之处,提出了两个扩展方法:一是使用模糊集理论扩展用例复杂度权重因子表,使权重因子能够更准确地反应事务数目的变化,从而使估算的结果更准确、可信;二是采用分解的方法使用例点方法能够实现对单个用例的估算,为项目计划提供必要的数据支持。扩展方法简明实用、灵活性强,易于软件组织实施和推广。

关键词:软件估算;用例点;模糊集;用例
中图分类号:TP311.52 **文献标识码**:A **文章编号**:1673-629X(2006)12-0064-03

Extended Use Case Point Estimation Method

ZHOU Yang, WU Hai-tao, ZHANG Dong-wei
(Mathematics, Science and Information College, Shanghai Normal University, Shanghai 200234, China)

Abstract: Analyses the shortcoming of the use case point (UCP) estimation method. Then two extensions of UCP was proposed, one extended use case complexity weight table uses the theory of fuzzy set, which makes weight factor be able to reflect the change of number of transaction more precisely; the other extension make UCP can be used to estimate the size of single use case by decomposition which can provide essential data support to project plan. The extended method is concise, practicable, flexible and easy to popularize.

Key words: software estimation; use case point; fuzzy theory; use case

0 引 言

软件估算是软件项目开发的一种活动,其目的就是通过对软件项目的规模、工作量、进度、关键计算机资源进行科学地预测,从而对项目开发做出严肃、合理的承诺,指导软件开发的整个过程[1]。捕捉软件项目的功能需求时,常使用用例模型[2]。既然用例已成为需求采集和分析的标准部分,那么,基于用例而不是采用功能点或代码行等其他技术,进行开发规模和资源的估算工作,是很有意义的。

用例点方法(use case point method)[3]是 Gustav Karner 在 1993 年提出的,这种方法是对 FPA 方法的改进,但又与 FPA 有着本质的不同。这种方法的基本思想是利用已经识别出的用例和执行者,根据它们的复杂度分类,计算用例点,然后利用用例点和工作量的换算,得到项目开发所需的以人小时数为单位的工作量。

大量的对用例点估算方法的研究表明,用例点算法是有效的,与专家估算相比较,两者的相对误差平方和的平均值几乎接近[4],该方法很可靠,至少像诸如 COCOMO、功能点和代码行等估算方法一样可靠[5]。

用例点方法是有效的且易于学习和使用,但用例点方法也存在一定的不足,下面的两个问题是很明显的:

* 在计算未调整用例权重(UUCW)时,如果有两个用例,一个用例的场景中的事务的数目等于 4,而另一个用例场景中的事务的数目等于 6,很明显后者的复杂度要比前者高,但在用例点算法中,两者对应的权重都是 10,这一点影响了估算结果的准确性。

* 项目管理中,单个或一组用例常被用作分配工作的基础,因此需要了解每一个用例的规模和工作量,以此作为项目计划的依据,目前用例点算法并不能完成对单个用例的估算。

对于以上涉及到的问题,文中将提出两个扩展方法来解决上述问题,第一个扩展通过使用模糊集合来解决问题 1,第二个扩展里通过把角色权重分解到各个用例实现对单个用例的估算。

1 用例点估算方法

用例点方法的使用步骤如下:

1) 计算总的未调整执行者权重(UAW)。

根据执行者与系统之间交互的复杂度,将其分为简单、中等、复杂三个类别,每一类别的权重因子分别为 1,2,3。项目总的未调整活动者权重(UAW)= $\sum_{i=1}^{n}$ 相应类别的活动者数量 * 权重因子。

2) 计算未调整用例权重(UUCW)。

根据用例场景中所描述的事务数来判断。事务定义为一系列的任务的原子集,这些任务要么一起完全执行,

收稿日期:2006-03-31
基金项目:上海市高等学校青年科研基金项目(CL200322)
作者简介:周 杨(1980-),男,江苏扬州人,硕士研究生,研究方向为软件工程;吴海涛,副教授,研究方向为软件工程和数据挖掘。

要么一起均不执行。分类的基础是用例中的事务数,包括扩展场景中的事务也必须计算在内。如表 1 所示。

表 1　根据事务分类的用例复杂度权重因子表

用例的类型	判断规则	权重因子
简单	事务的数目小于等于 3	5
中等	事务的数目在 4 到 7 之间	10
复杂	复杂事务的数目多于或等于 7 个	15

项目总的未调整用例权重(UUCW) = $\sum_{i=1}^{n}$ 相应类别的用例数量 * 权重因子。

3) 计算未调整用例点(UUCP)。

将 UUCW 和 UAW 相加,即可得到未调整用例点 UUCP。UUCP=UAW+UUCW。

4) 计算技术复杂度(TCF)和环境复杂度(EF)。

技术复杂度因素用来体现项目的非功能性需求对项目的影响,共包含 13 个因素。设 W_i 表示权重,R_i 表示相关度,TFactor = = $\sum_{i=1}^{n} W_i * R_i$,TCF(技术复杂度) = 0.6 + (0.01 * TFactor)。

环境因素列表的主要目的是利用项目所存在的风险来调整软件的规模,共包含 8 个环境因素。EFactor = = $\sum_{i=1}^{n} W_i * R_i$, EF(环境复杂度) = 1.4 + (−0.03 * EFactor)。

5) 计算调整用例点(UCP)和工作量。

利用 TCF 和 EF 对未调整用例点的影响可得到调整用例点,UCP = UUCP * TCF * EF。Karner 最初提出时,建议每个用例点的工作量为 20 人时,而 Ribu 在 Karner 的研究基础之上,认为这个工作量应该是在 15 至 30 个人时之间[5]。规模和工作量的换算为:工作量(人时) = 用例点数 * 每用例点工作量(人时)。

2　使用模糊集理论扩展用例点方法

2.1　模糊集定义

定义 1[6]:设在论域 U 上定义一个映射 $A: f_A(u): U \to [0,1]$,称 A 为 U 上的模糊集,$f_A(u)$ 称为 A 的隶属函数,或者称为 u 对 A 的隶属度。一般情形下 A 可以表示为:$A = \{(u, f_A(u)) \mid u \in U\}$

2.2　采用模糊集理论扩展用例点方法的理论基础和目的

模糊集理论在软件估算领域已有一定的研究,Osias de Souza Lima Júnior 在他的论文里提出了在功能点估算技术中采用模糊集合理论进行优化的一种方法[7]。这里将使用模糊集理论对用例复杂度权重表进行模糊处理,可以使权重能够更准确地反应事务数目的变化,目的是解决引言中提出的问题 1。

2.3　扩展方法

建立模糊集首先要选择隶属函数,一般采用参照法来表示属性值的隶属度。参照法是指利用现有的一些函数,通过参照比较,选择最能代表所讨论模糊集的函数作为隶属函数,在考虑原来的权重表分布特点的基础上采用梯形分布模型,$A = \{(u, f_A(u)) \mid u \in U\}$,$f_A(u)$ 为隶属度函数。

$$f_A(u) = \begin{cases} 0 & x < a \\ (x-a)/(m-a) & x \in [a,m] \\ 1 & x \in [m,n] \\ (b-x)/(b-n) & x \in [n,b] \\ 0 & x > b \end{cases}$$

$f_A(u)$ 的图形表示如图 1 所示。

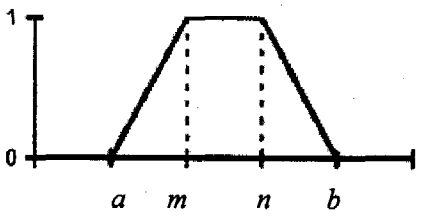

图 1　梯形化的模糊集

分两步说明如何使用选定的模糊集扩展用例点方法:

1) 模糊化关键词。

首先在用例复杂度权重因子表(见表 1)的基础上建立梯形化模糊集合。用例权重表中包括关键词集合 T_i("事务的数目小于等于 3"、"事务的数目在 4 到 7 之间"、"事务的数目多于或等于 7 个"),m_i 的值为相应关键词中事务数目的下限值,$n_i = (m_i + m_{i+1})/2$,$a_i = n_{i-1}$,$b_i = m_{i+1}$。经计算得到表 2。

表 2　模糊集参数表

m_1	1	a_2	2
n_1	2	b_2	7
a_1	X	m_3	7
b_1	4	n_3	X
m_2	4	a_3	5
n_2	5	b_3	X

2) 计算新的权重因子。

计算新的权重因子可以概括为以下两条规则,设 $P(u)$ 为事务数目为 U 的用例场景的权重:

* 当事务的数目 u 在相应模糊集合 A_i 的 m_i 和 n_i 之间时,权重因子 $P(u)$ 保持原来结果。

* 当事务的数目 u 在相应模糊集合 A_i 的 n_i 和 b_i 之间时,权重因子 $P(u)$ 的计算如下式:

$$P(u) = f_{A_i}(u) * V_i + f'_{A_i}(u) * V_{i+1}$$

其中: $f_{A_i}(u) = (b_i - u)/(b_i - n_i)$

$$f'_{A_i}(u) = 1 - f_{A_i}(u)$$

V_i 是相应的 T_i 在权重表中对应的权重因子($V_1 = 5$; $V_1 = 10$; $V_1 = 15$)。

举例:设某用例场景中事务的数目为 6,适用第二条规则,计算权重因子为:

$$P(6) = 0.5 * 10 + 0.5 * 15 = 12.5$$

以此类推,新的权重表设计如表 3 所示。

表 3 扩展后的场景权重因子表

事务的数目	权重因子
1	5
2	5
3	7.5
4	10
5	10
6	12.5
7 或多于 7 个	15

3 分解执行者估算单个用例

通过分解执行者权重的方法实现对单个用例的估算，设 $UUCP_i$ 表示第 i 个用例的未调整用例点，$UUCW_i$ 表示用例的未调整权重，$RUAW_i$ 表示与第 i 个用例相关的执行者权重。设有 n 个执行者是该用例的主执行者，其中每一个执行者可能是 m 个用例的主执行者，则与第 i 个用例相应的未调整执行者权重($RUAW_i$) 可以计算得到：

$$RUAW_i = \sum_{j=1}^{n} UAW_j * \frac{UUCW_i}{\sum_{k=1}^{m} UUCW_k}$$

其中 $\sum_{k=1}^{m} UUCW_k$ 是 UAW_j 所代表的执行者作为主执行者的所有用例场景的权重和，则第 i 个用例的未调整用例点 $UUCP_i = RUAW_i + UUCW_i$。

4 案例研究

4.1 估算结果

结合几个项目的实际数据，对几种估算方法进行了对比，表 4 是这几个项目的估算和结果比较。

表 4 项目估算表和估算结果比较

项目	编号	实际值(人时)	用例点(人时)/错误率	专家经验(人时)/错误率	扩展用例点(人时)/错误率
统计局业务系统	SHSIS	10994	7380/32.87%	8056/26.72%	7421/32.5%
Lucky ERP System	LuckyERP	10336	8642/16.39%	8816/14.7%	8428/18.46%
办公自动化系统	MHOA	6688	6240/6.7%	8080/20.81%	6459/3.4%
房产交易系统	HTS	2972	4428/49%	3800/27.86%	4158/39.91%

4.2 估算结果分析

* 对比专家经验估算和用例点估算。

4 个项目中有两个项目专家经验的错误率比用例点方法低 5 个百分点以上，有一个专家经验比用例点方法高 10 个百分点以上，有一个项目两种方法大致相等。总体而言，用例点估算方法是一种有效的估算方法，可以和专家经验结合使用，当两种估算方法结果偏差较大时可以考虑重新估算。

* 对比扩展用例点估算和用例点估算。

扩展用例点方法和用例点方法估算的结果比较接近，这是由于扩展用例点方法是源自用例点方法的。在 4 个项目中有 3 个项目扩展方法的错误率比用例点方法低，而且从 4 个项目综合来看，扩展用例点方法也比用例点方法具有略高的准确度。所以，在使用用例点方法的场合，可以使用扩展用例点方法的用例复杂度权重因子表(见表 3)来替代用例点方法中的权重因子表(见表 1)，以获得更高的准确度。

* 估算单个用例。

在单个用例的估算上，对比专家经验和扩展用例点方法可以发现两者的差异与两者对项目整体估算的差异是大致相等的。所以，扩展用例点方法对单个用例估算和对项目整体进行估算具有基本一样的准确度。

5 结 论

文中的主要研究成果在于在研究估算技术、用例、用例点估算方法和模糊集的基础上提出了两个用例点估算方法的扩展：一是使用用模糊集扩展用例点方法，扩展后的方法将比用例点方法具有更好的准确度，并且不会使原用例点方法变得复杂；二是通过把角色的权重分解到各个用例，从而实现对单个用例的估算。经过与专家经验以及实际值的比较，发现对单个用例的估算的准确度和对整个项目估算的准确度是大致相等的，这说明扩展用例点方法对单个用例的估算是有效的、可靠的。

参考文献：

[1] Kishore S. 软件需求与估算[M]. 北京：机械工业出版社，2004.

[2] Jacobson. Object - Oriented Software Engineering: A Use Case Driven Approach[M]. New York: Addison - Wesley Publishing Company, 1992.

[3] Karner G. Metrics for Objectory[D]. Sweden: University of Linköping, 1993.

[4] Ribu K. Estimating Object - Oriented Software Projects with Use Cases[D]. Norway: University of Oslo, 2001.

[5] Damodaran M, Washington A N E. 用例点估算方法[J]. 非程序员，2005(45)：61 - 65.

[6] Zadeh L A. Fuzzy Sets[J]. Information and Control, 1965, 8: 338 - 353.

[7] de Souza Lima Júnior O, Farias P P M, Belchior A D. A Fuzzy Model for Function Point Analysis to Development and Enhancement Project Assessments[J]. CLEI Electronic Journal, 1990, 5(2): 1 - 14.

(上接第 63 页)

2002.

[2] Flenner R, Abbott M. Java P2P 技术内幕[M]. 北京：人民邮电出版社，2003.

[3] Boger M. Java 与分布式系统[M]. 北京：机械工业出版社，2003.

[4] 朱文华，王 茜. 基于 Jini 技术和 JavaSapce 构建分布式工作流管理系统[J]. 计算机应用研究，2003.(6)：82 - 84.

[5] Newmarch J. A Programmer's Guide to Jini Technology[M]. [s.l.]: Apress, 2000.