文章编号:1001 - 9081(2007)05 - 1248 - 03

一种 XP 项目迭代周期估计方法

王晓华[1,2],赵 明[1]
(1. 贵州大学 可靠性工程研究中心,贵州 贵阳 550025; 2. 遵义医学院附属医院,贵州 遵义 563003)
(gzzywxh@hotmail.com)

摘 要:XP 项目开发周期依赖于项目团队的开发效率。这个效率可以用每个迭代开发阶段加入的用户 Story 被成功实现的概率来描述。提出了一种简单、实用的估计 XP 项目开发效率和迭代周期的数学方法。该方法可用于项目开发成本和工作量估计。用一组数据对该方法进行了模拟验证。

关键词:极限编程;用户 story;开发效率;迭代周期
中图分类号: TP311.52 **文献标识码**:A

An estimation method of iteration periods in XP project

WANG Xiao-hua[1,2], ZHAO Ming[1]
(1. *Reliability Engineering Center, Guizhou University, Guiyang Guizhou* 550025, *China*;
2. *Affiliated Hospital of Zunyi Medical College, Zunyi Guizhou* 563003, *China*)

Abstract: The development period of a Extreme Programming (XP) project depends on the productivity of development team. The productivity can be described by probability of which all user stories are achieved successfully in each iteration phase. In this paper, we presented a simple, but realistic mathematic method to estimate the productivity and iteration periods of XP project. The approach can be useful in estimation of development cost and effort. The approach was verified valid with a set of simulation data.

Key words: extreme programming; user story; development productivity; iteration periods

0 引言

极限编程(Extreme Programming,XP)是主要的敏捷软件开发方法,在近十年时间里为大家所熟悉。这是一种适合于中小规模项目的轻量级开发方法。有关极限编程的研究文献主要集中在实际应用及与传统开发方法的比较方面,关于其度量的研究极为少见。Marco Melis[1]等对 XP 开发过程进行数学模拟以评估 XP 实践的效率, M. Muller[2]等探索了 XP 方法高开发成本及短上市时间和高代码质量之间的关系。Succi, G 和 Stefanovic, M[3]等提出了结对编程成本效益的试验框架。目前还未见评估 XP 项目开发效率及估计 XP 项目迭代周期的计算方法。

本文的研究目标在于促进对 XP 项目用户 story 管理过程的理解和应用,并通过其估计开发效率和迭代时间。在 XP 实践中,用户需求被分解成具体的、明确的用户 story,这些 story 能够被估计出开发时间,有相应测试案例,适当的大小(一个 story 的完成时间,也叫 ideal days/story points)。在 XP 项目实施过程中,我们可以得到每个迭代阶段成功实现及未完成的用户 story 数,我们可以用它们来度量项目团队的开发效率及估计项目开发时间。本文提出了基于用户 story 管理过程的 XP 项目开发效率和迭代周期的估计方法。

1 XP 中用户 Story 处理过程

在使用 XP 方法的项目中,开发人员每次开发时总是完成最主要的功能,然后不断在下一次迭代或下一个版本中加入新的 story。在极限编程实践中,一个软件项目的规模实际上可用 Story 数来度量,还可以根据每个迭代周期长短和迭代次数来计算其迭代工作量。

由于在每个迭代阶段有一些用户 story 能在一个迭代周期内完成,而另一些却不能,因此在不同的阶段会投入不同数量的 story。在实际开发活动中,不同的团队对每个 story 需要的工作量大小划分标准不一样,每次迭代要完成的 story 数也不同,不同团队每一次迭代所分配的工作量也不一样,一般一个迭代周期在 1 ~ 3 周之间,或者更长时间,这依赖于系统规模和系统复杂性[4]。然而,在一个具体项目中,可以保证迭代周期一般是一个固定值,每个 story 大小基本一样。

在一个迭代阶段将会有一部分 story 被延续到下一阶段完成。出现这种情况一般有四种原因:(1)开发人员经验不足或者对领域知识了解不够,使得 story 定义范围过大,不能在一个周期实现其规定的需求,需要再细分;(2)程序员完成该 story 的速度低于预期速度,需要延期;(3)已完成的 story 还存在错误,如果改正这个错误需要一个工作日以上,则改正错误也被看着是一个被延续下来的 story;(4)已完成的 story 偏离用户要求,需要重做。当然,也会有一些不成熟的 story 会在开发过程中被客户要求抛弃。

图 1 是 XP 项目的用户 Story 被迭代处理的示意图,也即用户需求的动态处理过程。在每次迭代开始之前,都要举行计划(Planning game)会议。客户提出需求,双方再共同划分 story,开发人员需要考虑每个 story 是否可以进行测试,要能够估计合适的大小,以便于实现,如果不能满足这两点要求,开发人员需要用户阐述清楚或者重新划分[5,7]。每个迭代阶段开发人员都要编写测试用例和代码,运行软件,并测试,一

收稿日期:2006 - 11 - 21;**修订日期**:2007 - 02 - 07 **基金项目**:教育部春晖计划资助项目(S2005 - 2 - 52001)
作者简介:王晓华(1970 -),男,贵州遵义人,博士研究生,主要研究方向:软件工程、软件可靠性与安全性; 赵明(1963 -),男,河南驻马店人,教授,博士生导师,博士,主要研究方向:软件工程、可靠性工程.

次迭代就是一个小的增量过程。在 XP 项目的第一个迭代阶段完成以后,有一部分未完成的用户 Story 将呈前面所述的 4 种变化,被延续到下一个迭代阶段。在新的迭代阶段,通过开发人员和用户共同商定加入新的 Story,以满足一个迭代周期需要的工作量。一部分 Story 已经成功完成表明软件的某一部分功能已经得以实现,已经作为软件产品的一部分,这样通过逐次迭代添加,最后产生一个完整的软件产品,可以进入集成测试阶段。

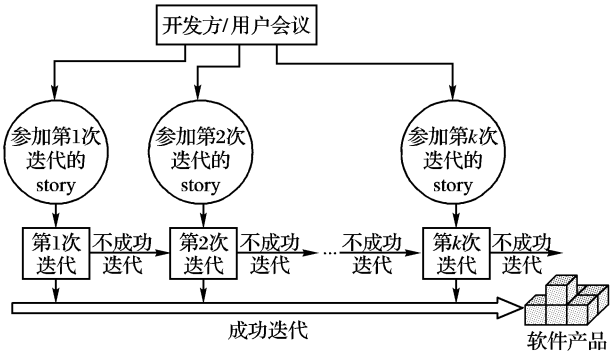

图 1　XP 项目用户 Story 迭代过程

文献[6]把软件开发任务的实施或者软件错误的修改看成是以一定概率进行的,因此我们也认为每个阶段的用户 story 总是以一定的概率 p 能够被完成,则另一部分以 $1-p$ 的概率延续到下一阶段。而 p 值正好可以视为用户 story 的完成效率。在新的迭代阶段,总的 story 由新加入的 story 和上一个阶段的遗留 story 组成。对于上一阶段未完成的 story,由于上一个阶段已经做了一些工作,多数不再是一个完整的 story(从一个 story 规定的工作量上讲),我们将把他们折合成具有规定 story 大小的 story 数,统称为上阶段遗留 story。这样,我们可以搜集各阶段的已实现 story 数和遗留 story 数,当项目进行到某个阶段时,通过本文提供的模型估计出 p 值,并进一步估计出迭代周期。

一个好的用户 story 应该易于理解,在规定的迭代时间内能实现。这依赖于与客户的充分的沟通,合理的问卷,敏锐的观察,正确的书写等能力[7]。用户 story 的质量直接影响 p 值的大小。

2　估计 p

本文估计方法基于下面的假设:开发人员在开发过程中保持不变,不增加和减少人员,不更换人员,保证其有相同的开发速度;项目管理人员有一定的 XP 项目经验,对项目的初始规模(story 数)把握较为准确;每个用户 story 的大小基本相同(对于大小不同的 story,可以通过对用户 story 分解或组合来实现),即完成每个 story 需要的工作量大致相同。

假设 XP 项目进行到在第 i 个迭代阶段初期,我们用 f_i 表示第 $i-1$ 阶段的遗留 story 数,用 m_i 表示第 i 阶段新引入的 story 数。显然,在第一个迭代阶段,$f_1=0$。则在第二个迭代阶段初,上一阶段遗留的 story 数 f_2 与 p 和 m_1 的关系如下:

$$f_2 = m_1(1-p) \tag{1}$$

第二阶段需要完成的 story 数为 $m_1(1-p)+m_2$,其中 m_2 为第二阶段引入的新 story 数。

各迭代阶段的 story 构成见表 1。在每个阶段,上阶段的遗留 story 数和本阶段加入的 story 数都是可以直接从项目实施过程中获得的,因此当项目发生几次迭代以后,我们可以通过这两种数据估计出 p 的值。

表 1　不同迭代阶段的用户 story

迭代阶段	遗留story	新加入story	总投入story
1	$f_1=0$	m_1	m_1
2	$f_2=m_1q^*$	m_2	f_2+m_2
3	$f_3=f_2+m_2q$	m_3	f_3+m_3
…	…	…	…
$k-1$	$f_{k-1}=q(f_{k-2}+m_{k-2})$	m_{k-1}	$f_{k-1}+m_{k-1}$
k	$f_k=q(f_{k-1}+m_{k-1})$	m_k	f_k+m_k

*$q=1-p$

假设在项目完成第 $k-1$ 次迭代后,进入到第 k 次迭代,这时要去估计 p 值。可知从第 1 个迭代阶段到第 $k-1$ 个阶段遗留下来的用户 story 数 $f_j(j=1,2,\cdots,k)$ 和每次加入的 story 数 $m_i(i=1,2,\cdots,k-1)$。从表 1 可以得到公式:

$$q = 1-p = \frac{\sum_{j=2}^{k} f_j}{\sum_{j=2}^{k-1} f_j + \sum_{i=1}^{k-1} m_i} \tag{2}$$

和公式

$$\sum_{i_1=1}^{k-1} m_{i_1} q + \sum_{i_2=1}^{k-2} m_{i_2} q^2 + \cdots + (m_1+m_2)q^{k-2} + m_1 q^{k-1} = \sum_{j=1}^{k} f_j, \quad q = 1-p \tag{3}$$

在上式中,m_i 和 f_j 已知,因此可以根据前面若干次(大于等于2次,因为第一次迭代不存在上阶段遗留 story,即 $f_1=0$)迭代获得的实际数据,通过公式(2)或(3)计算 p 的估计值。

3　估计迭代次数

尽管 XP 项目不主张在项目初期实现所有需求分析,但在实际开发中大致了解一下总体需求,并初步估计项目的规模,是必要的。因而在做完项目可行性分析和初期需求分析后,根据开发经验,可以大致给出项目的估计 story 总数,设为 N。

设总的迭代次数为 g 次,则在第 g 次迭代后,用户将不再增加新的 story,迭代会结束,这时 $m_{g+1}=0$。但第 g 次迭代后仍然会有遗留的 story 数

$$f_{g+1} = (f_g + m_g)(1-p)$$

要在 $g+1$ 阶段迭代最后完成,实际迭代次数可以为 $g+1$。我们也可以视这一部分遗留的用户 story 为在一个延时的迭代阶段完成。

根据表 1,可知平均每个阶段引入新的 Story 数为:

$$\bar{m} = \frac{\sum_{i=1}^{k-1} m_i}{k-1} \tag{4}$$

还可以得到

$$m_{k-1} = \frac{f_k}{1-p} + f_{k-1} \tag{5}$$

其中 f_{k-1}、f_k 分别是第 $k-2$ 次、第 $k-1$ 次迭代遗留的 story 数。根据(2)式有:

$$\bar{m} = \frac{\frac{p}{1-p}\sum_{j=2}^{k-1} f_j + f_k \frac{1}{1-p}}{k-1} \tag{6}$$

则从 N 个 Story 迭代的可能次数为:

$$g = \frac{N}{\bar{m}} \tag{7}$$

(6)式代入(7)式,得:

$$g = \frac{N(k-1)}{\frac{p}{1-p}\sum_{j=1}^{k-1} f_j + \frac{f_k}{1-p}} \quad (8)$$

可以看出迭代次数 g 可以通过 p 以及每个阶段遗留的 story 数 f_j 来估计。如果某 XP 项目规定一个迭代周期为 7d，则团队在迭代阶段的工作总量为 $7 \times (g+1)$d。

4 一个样例

表 2 中 $k = 4$，说明项目刚进行到第四个迭代阶段，前 3 次迭代已经完成。我们将用前 3 次迭代所得数据去估计 p，并预测总的迭代次数。

表 2 一个用户 story 处理过程(数据)示例

迭代阶段	上阶段遗留(未完成)story	本阶段新增story	本阶段需要完成的story
1	0(0)	28	28
2	7(12)*	22	29
3	9(13)	23	32
4	8(12)	17	25

*括号中的数字表示未完成的story数，
括号外的数字表示折合工作量后的遗留story数

如表 2 所示，$m_1 = 28, m_2 = 22, m_3 = 23, f_1 = 0, f_2 = 7, f_3 = 9, f_4 = 8$。应用(2)式，解得 $p = 0.7303$。应用(3)式，解得 $p = 0.7291$。两种计算方法计算出的 p 值非常接近。其中(2)式更便于计算。不难看出，k 越大，获得的数据越多，p 值越趋于稳定，但 k 取得太大则失去预测的意义。

取 $p = 0.7303$。我们使用在文献[2]提供的一个 XP 项目，其 story 总数为 $N = 168$。再应用(8)式得可得 $g = 6.9053$。

即对于初步估计 story 总数为 168 的软件项目，使用表 2 提供的经验数据，在开发人员不变的情况下(以保持同样的 p)，采用 XP 方法大约需要经历 7 个迭代阶段才能把用户需求完全加入到入软件产品中。由于第 7 次迭代后还有遗留 story，可以认为总的迭代次数为 8 次。如果计划每 3 次迭代进行一次小版本发布，则该项目需要 3 次版本发布。随后再进行进一步的集成测试，试运行和验收。

完成了对 XP 项目迭代周期和发布次数的估计，很容易进一步做出整个项目开发时间的预测，并做出资源分配和成本估算。

5 结语

以上提供一个简单的 XP 项目迭代周期估计模型，其用于计算的数据在项目实施过程中很容易获得，使得用项目早期开发数据来估计整个项目开发周期更容易实现，能够得到较为准确的数据，对进一步做好项目资源分配和成本估算有重要参考价值。其中 p 值也可以作为 XP 项目团队开发效率评价的参考数据。p 值的估计充分利用了项目开发过程中用户 story 被处理情况的数据，因而具有较强的代表性。

在 XP 项目里，如何尽早发现隐藏的“坏味道”(Bad smell)，保证用户 story 易设计、可测试、大小合适，是影响项目开发进度的重要环节。一个好的用户 story 将减少重构次数，从而提高开发效率。用户 story 的重写及代码的重构过于频繁，是延缓 XP 项目的主要原因。XP 项目管理者还可以进一步搜集本文所述 4 种 story 延迟数据，做进一步分析，以改进软件项目需求变更管理，促进需求分析质量和开发质量的提升。

参考文献：

[1] MELIS M, TURNU I, CAU A, *et al*. Evaluating the impact of test-first programming through software process simulation[J]. Software Process: Improvement and Practice, 2006, 11(4): 345 - 360.

[2] MATTHIAS MM, PADBERG F. On the economic evaluation of XP project[A]. ESEC 9[C]. 2003. 168 - 177.

[3] SUCCI G, STEFANOVIC M, PEDRYCZ W. Quantitative assessment of extreme programming practices[J]. Canadian Conference on Electrical and Computer Engineering, 2001, 1: 13 - 16.

[4] MCGOVERN F. Managing software project with business-based requirement[J]. IT Professional Publication Date, 2002, 4(5): 18 - 23.

[5] BECK K, FOWLER M. Planning Extreme Programming[M]. Addison-Wesley, 2001. 50 - 52.

[6] XIE M, ZHAO M. Modeling and estimation of MTBF in a Test-Analyse-And-Fix Environment[A]. Reliability&Quality In Design - Proceedings of the ISSAT International Conference[C]. 1994. 399 - 401.

[7] COHN M . User Stories Applied for Agile Software Development [M]. Pearson, 2004. 73 - 84.

(上接第 1237 页)

贝叶斯模型的强独立性假设和 EM 算法对于初始数据值的依赖性，往往得到的分类效果不能令人满意。在本文中，我们提出了一种基于不完整数据集的改进分类方法(B-EMNB)。它利用与朴素贝叶斯文本分类相同的思想，通过引入 Bernoulli 混合模型和带有权值的 EM 算法依靠最大似然函数估计得到分类器的参数。随后的实验表明，在数据量充足的情况下我们的方法能够更好的改善分类效果。进一步的工作主要包括如何对样本集的特征值提取和选择方面做出改进，来增强分类器的精度。

参考文献：

[1] RONALD R, YAGER. An extension of the naive bayesian classifier [J]. Information Sciences, 2006, 176: 577 - 588.

[2] RAMONI M, SEBASTIANI P. Robust bayes classifiers[J]. Artificial Intelligence, 2001, 125(2): 209 - 226.

[3] QUINLAN JR. C4. 5: Programs for machine learning[M]. San Francisco, CA: Morgan Kaufmann, 1993.

[4] KAMAL N, ANDREW KM, SEBASTIAN T, *et al*. Text classification from labeled and unlabeled documents using EM[J]. Machine Learning, 2000, 39: 103 - 134.

[5] TSURUOKA Y, TSUJII J. Training a naive bayes classifier via the EM algorithm with a class distribution constrain[A]. Seventh Conference on Natural Language Learning[C]. Edmonton, Alberta, Canada: Walter Daelemans and Miles Osborne, 2003. 127 - 134.

[6] JUAN A, VIDAL E. On the use of Bernoulli mixture models for text classification[J]. Pattern Recognition, 2002, 35(12): 2705 - 2710.

[7] HAN JW, KAMBER M. 数据挖掘: 概念与技术[M]. 范明, 孟小峰, 译. 北京: 机械工业出版社, 2003.

[8] MITCHELL TM. 机器学习[M]. 曾华军, 张银奎. 北京: 机械工业出版社, 2003.

[9] BLAKE C, KEOGH E, MERZ C. UCI repository of machine learning database[EB/OL]. http://www.ics.uci.edu/mlearn/ML Repository.Html, 2006 - 09 - 05.