基于敏捷方法的软件项目管理研究*

关忠诚,程 刚
(中国科学院 研究生院管理学院,北京 100049)

摘要:本文首先探讨了敏捷项目管理的起源及其适应性项目框架;并论述了其在软件项目中的应用。然后对适应性项目框架的计划制定对比极限项目管理作了详细的阐述。

关键词:敏捷项目管理;适应性项目框架;极限项目管理
中图分类号:F273 **文献标识码**:A **文章编号**:1008-5831(2005)06-0046-04

A Study on the Software Project Management Based on the Agile Project Management

GUAN Zhong-cheng, CHENG Gang
(*School of Management, Graduate University of Chinese Academy of Sciences, Beijing 100049, China*)

Abstract: The origin of the concept of agile project management is discussed in this article, as well as the framework of adapted project. The application of the agile project management among the software projects is also talked about. Compared with the extreme project management, the planmaking of agile project framework is elaborated in detail.

Key words: agile project management; framework of adapted project; extreme project management

一、引言

软件开发中既有高风险、高变化的项目,也有目标明确、解决方案明了的低变化项目。根据不同的项目特点,选用不同的项目管理方式是项目成功的关键。敏捷项目管理是应对经常变化的、具有不确定性的软件项目的管理方法。敏捷即灵活性,是动态的、适应于具体情况、迎合变化和自我完善的。本文针对敏捷项目管理中的极限项目管理和适应性项目框架的软件应用对比传统项目管理进行探讨,并提出了适应性项目框架的改进和计划控制的一些建议。

二、敏捷项目管理的概念及起源

敏捷项目管理的概念来源于敏捷软件开发。随着敏捷软件开发的发展,极限项目管理(也称为极端项目管理 Extreme Project Management 或 Radical Project Management)和敏捷项目管理(也称为灵活的项目管理 Agile Project Management)的概念和方法被相继提出,并仍在不断发展。实际上,敏捷项目管理只是各种敏捷软件开发方法相应项目管理的统称,只针对于软件项目,并不是一种通用项目管理方法(也有人提出敏捷项目管理的通用概念,但未被广泛接受)。极限项目管理和适应性项目框架皆源于对 Doug DeCarlo 于2000年发布的弹性项目模式(Flexible Project Model)的改编。而弹性项目模式又来自于敏捷软件开发中的自适应软件开发方法学的启发。现在二者已经发展成为一个通用的项目管理理论。极限项目管理适合于变化大、复杂程度高的项目。传统的项目管理则适合低变化、低不确定性的项目。而在二者之间是适应性项目框架。虽然所有的敏捷软件开发方法都被认为是属于极限项目管理的范畴,但从最近的敏捷软件开发的发展可以看出有些敏捷方法并不全属于极限项目管理的范畴。而且极限项目管理往往由于过于激进,显得不够实际,并不能被高级管理者特别是 CIO 所接受,且在大型项目中也无法得到有效论证。现在的敏捷项目管理研究大多有转向适应性项目框架的趋势。所以,虽然敏捷项目管理通常指的就是极限项目管理,但它被认为应是包括极限项目管理和适应性项目框架两部分的软件项目管理的统称,极限项目管理又是适应性项目框架的特例。

三、敏捷项目管理的适应性项目框架

通用适应性项目管理框架是以客户为中心、客户

* 收稿日期:2005-10-10
作者简介:关忠诚(1965-),男,山东人,中国科学院科技政策与管理科学研究所研究员,主要从事管理科学、科技评价研究。

驱动的管理方法。极限项目管理是处在比适应性项目框架更复杂,更不确定的高变化情况下的一种管理方法。二者区别在于,适应性项目框架是针对有明确的目标但没有解决方案的项目,而极限项目管理则是针对两个方面都很模糊的情况下的探索式的方法。适应性项目框架只要求客户在每个迭代周期的实施结束后参与项目,而不是全程参与到项目中。

适应性项目框架主要分为定义项目范围、制定项目周期计划、项目实施、客户检查、项目后回顾五个阶段(图1)。

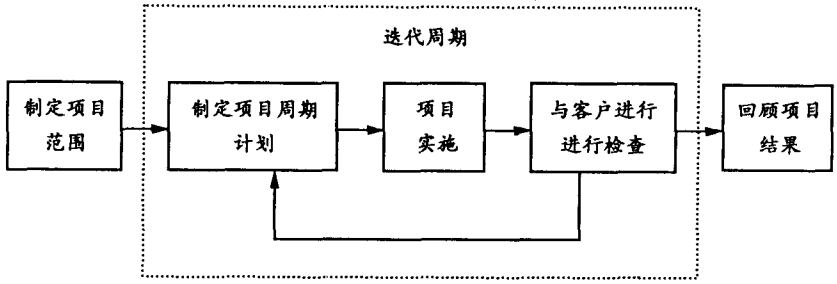

图1　适应性项目框架主要阶段

其中项目范围包括项目满意条件、项目概况说明书、功能要求优先排序、中层 WBS 等。中层工作分解结构只是分解到功能级,而不是任务级,只要可以比较确信地估计每一段功能所需的时间和资源就已足够。因为对于经常变化无法预计的任务,编写完整的 WBS 完全是浪费。制定项目周期计划是将要进行的下一个周期的详细计划,是带有依赖关系的任务层次的详细实施计划。项目实施阶段包括制定微观进度计划、实现功能、监督并调整实施进度。在这个阶段,取消当前周期和调整计划都是可以执行的,可以减小和避免损失。通过中间三个阶段的反复进行,最后可以实现客户满意的解决方案。

然而适应性项目框架并没有指出当项目出现变化时,如何在时间成本有限的情况下有效完成任务。它还是沿用极限项目管理的方法,制定详细的周期计划,必要时抛弃部分要完成的功能。区别是增加了中层的项目计划。中层项目计划根据时间限制范围内,能够容纳多少迭代周期,并根据特定周期内子功能的数量和质量调整周期时间。虽然也有风险分析,但是并没有在框架内体现并整合到项目周期中。如果根据这个计划确定项目的交付日期,则当变化发生时,很容易陷入传统项目管理的困境,即使采用迭代过程,也很难按期交付。迭代过程唯一能做的就是使变化或风险提前出现,而在以后的迭代周期改进,通常赶进度的方式是加班或者增加资源,这会使成本增加。而且质量的改进也是此类项目的一个不确定因素,这也需要时间和成本。如果低质量的产品延续到项目后期,则由于变化产生的时间和成本的消耗可能是致命的,也会增加维护的成本。适应性项目框架并没有考虑质量改进过程,而且忽视了初始迭代周期的作用。初始迭代周期完成后是调整计划的最佳时期,因为它是实际情况的真实体现,即使以后的迭代周期的实际情况会和初始周期有所偏差,但也不会过于偏离,而且随着迭代的进行,不确定性会减少。所以好的计划是使收益得到保障的首要因素。对适应性项目框架的软件应用改进主要是在过程中强调了风险管理和质量管理,并修改了计划部分,并着重强调了初始周期的作用。影响此类项目完成的最大的风险是需求变更造成的返工成本、时间消耗,需要靠风险缓解和质量控制来共同管理。所以,改进的重点在于适应需求变更。

软件开发项目的适应性项目框架如图2。图2中主要增加了风险缓冲后的基准计划、功能需求变更周期和质量改进周期。功能需求变更周期和质量改进周期是历时比较长的足够影响进度的活动。功能需求变更周期是由于业务需求变化导致的,可能是特定功能完全重新实施或者改进的过程。质量的改进周期区别于在功能需求变更周期的工作,这是在一定时期后,内部人员根据已完成项目功能的学习和经验总结进行的重新设计,改进已完成工作的质量,或为了适应变化所做的技术改进。风险缓冲后的基准计划是在中层计划基础上增加了风险缓冲的时间,包括功能需求变更周期和质量改进周期的预估时间。分离实施周期和修改周期是因为实施周期的时间和成本的预估是比较准确的,但修改周期的时间和重复次数却难以预测。在两个迭代周期的外围是个质量改进周期,表明在多个功能周期后进行质量改进。在改进前,需要评估改进的风险,作出权衡。这个周期结束后的产品被认为是一个非完全功能的发布版本。每个迭代周期还是和适应性项目框架中的一样,包括周期计划、实施和客户检查。另外一个区别是适应性框架只要求客户在客户检查点上参与,而这里要求全程参与,至少应在项目的前期阶段全程参与需求分析,目的是在一定程度上稳定需求,如果已经完成的功能

再出现需求修改,付出的成本和时间将会大得多。如果出现了范围的变更则和客户协商调整基准计划。

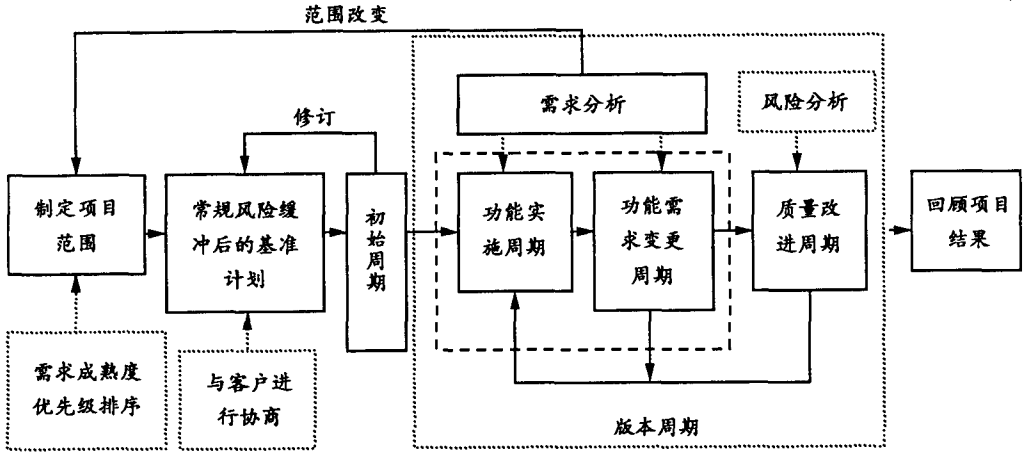

图 2 软件开发项目的适应性项目框架

四、基于敏捷方法的软件项目管理的计划与实施

(一)项目计划与风险

由于项目过程由传统的详细的需求计划的单一过程变成短的时间区间的具有反馈的多次迭代过程,并且为了对变化具有适应性,敏捷项目管理的计划方法分成详细周期计划与风险计划和质量计划结合的两种分层计划。

项目中的风险分为两种,一种是必然要发生的常规风险,一种是不确定的致命风险。前者可以通过风险缓解解决,后者则需要风险缓解和风险转化共同解决。在变化性比较强的软件项目中,需求变更必将发生,这是软件项目所面临的主要风险,计划中加入需求变更周期的缓冲时间可减小项目的成本风险。

(二)极限项目管理计划与时间预测

极限项目管理是一个无基准计划的过程,没有时间和成本的限制,利用数量不定的短周期不断迭代,最终完成项目,或者在完成前,项目就被取消。传统的项目管理计划方法在此时起不到太大的作用,一般采用跟踪团队的开发速度和剩余的功能点来进行管理,只制定迭代周期内实施的计划。

图 3 是 Burn - down 图,显示的是每个月后剩余的功能点数目。虚线 1 是要在 12 月完成的项目的计划线,虚线 2 是实际工作后的趋势线,这表明可能完成的日期。2 月剩余的功能点多于初始点,是因为增加了新的功能。可以看到这并不能确定交付的日期,只能作为参考。

(三)基于敏捷方法的软件开发计划

极限项目管理在某些极端的情况下确实很有效,比如新技术产品的研发。但是大多数软件项目的技术复杂度不是很高,或者曾有过类似的项目经验,项目的主要不确定性是业务需求的变化等,并受到时间

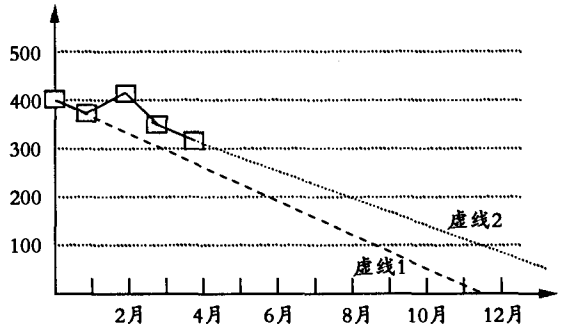

图 3 Burn - down 图

和成本的限制,这些项目的重点是得到反馈并改进。在这种情况下还需要预估时间,并制定相应的计划实施。软件项目由于成本与开发人员的多少和开发持续周期密切相关,所以时间通常是软件开发项目的一个重要衡量指标。使用改进后的自适应项目框架,利用需求优先级的功能排定和需求成熟度分析进行时间缓冲,可以很好地适应这种状况。

1. 需求的优先级

根据客户的要求排定功能的优先级。将有则更好,没有也不影响系统实用性的功能放在最后实现,是时间紧迫项目通常采用的方法。在进度延期的情况下,舍弃这些浮华的功能,可以确保项目按期完成,也可避免项目因增加了一大堆客户要求的功能而陷入遥遥无期的悲惨境地。

2. 需求的成熟度(见表)

需求的成熟度指的是需求的稳定程度。由于软件项目的范围通常是变化的,与客户洽谈并确定各个功能的需求成熟度并列表,既可以了解客户对新业务需求的理解程度,也是在需求变更时和客户谈判的依据。需求的成熟度越高,需求的变化程度越低,对需求变化的缓冲也就越小。变化的时间损耗指的是相对于功能实施时间的百分比。因为对原有功能的变

更通常会利用已有的模块,相对时间较短。变化的次数是指决定变更迭代周期的实施次数。

表　成熟度和变化缓冲因子的对应关系

需求的变化性	需求的成熟度	变更的时间损耗比	变化的次数
高	极低	0.5	3
较高	低	0.4	2
中等	中等	0.35	1
低	较高	0.2	1
极低	高	0	0

注:表中的具体数值可以根据项目的实际情况确定。

3. 需求缓冲的计算

变化的权值 = 变更的时间损耗比 ×变化的次数;

功能的需求变更时间缓冲 RAT = 功能的实施时间× 变化的权值;

迭代周期的需求变更时间缓冲 $CAT = \sum_{i=1}^{n} RAT_i$;

每个功能变化的权值为变更的时间损耗比乘以变化的次数。用此功能的实施周期预估时间乘以变化的权值则得出每个功能的需求变更的可能损耗的时间。将每个迭代周期内所包含的功能的需求变更时间相加,则得出了此迭代周期的需求变更时间,即需求变更周期的时间之和。

4. 计划的制定

计划时首先确定出项目范围后,创建出中层 WBS(工作分解结构),并以此确定项目功能的优先级,并根据风险分析确定每个功能的变化权值。确定好每个迭代周期时间,并根据总时间、成本的限制和功能优先级制定好功能的实施迭代周期数量及相应完成的功能制定计划。将功能需求变更周期和质量改进周期也考虑到计划中。由功能变化权值、功能实施周期时间和迭代实施周期内的功能数量决定的总的功能需求变更周期的时间,将时间缓冲合并入计划中。确定功能周期及修改周期重复的次数定期进行质量改进,一般是3到6次功能周期后进行一次质量改进,改进的具体内容则是实时制定。如果采用的是全程客户参与的方法,则需求成熟度会随着项目的进行趋于稳定,后期实现的功能可不设定需求变化的缓冲时间。由于大多数软件应用开发的变化没有在极限项目中的剧烈,即使增加了新的功能,修改非任务级的中层计划也比较容易,而且变更只在用户检查阶段实施。大多数情况下,在项目缓冲允许范围内的延迟不需要调整计划。Burn-down 图可作为一种辅助的管理方法。在短的时间周期内(一般是两周到五周),项目可以认为是低变化的,所以可以制定好将要进行的下一个迭代周期的详细计划。周期详细计划由于时间周期比较短,可不用关键链法。

由于软件开发项目的大多数模块都可以并行开发,并行的限制只受限于开发人员的数目,并且周期的时间都很短,所以在周期详细计划中使用关键路径并不能收到预期的效果,在项目实施中可应用的是关键链的项目缓冲机制。

由于敏捷项目管理的迭代性,根据已执行完的第一个迭代周期或前几个周期,可测量出比预估更有效的开发速度,可判定预估的准确度,从而调整基准计划。

五、结束语

敏捷方法在软件项目管理的采用确实提高了软件开发项目的成功率。但研究刚刚开始,仍然处在未成熟阶段。如何对经常变化的不确定性软件开发项目进行有效的敏捷管理仍是一个需要完善的课题。

参考文献:

[1] SANJIV AUGUSTINE, SUSAN WOODCOCK, Agile Project Management[EB/OL]. www. ccpace. com, 2003 - 06 - 10.

[2] ROBERT C MARTIN. PERT, CPM, and Agile Project Management[J]. Software development, 2004,(10):47 - 50.

[3] 罗伯特·K·威索基,等. 有效的项目管理(第三版)[M]. 北京:电子工业出版社,2004.

[4] 罗布·托姆塞特. 极限项目管理[M]. 北京:电子工业出版社,2003.

[5] DAVID J ANDERSON, ELI SCHRAGENHEIM. Agile Management for Software Engineering: Applying the Theory of Constraints for Business Results[M]. Upper saddle River, New Jersey:Prentice Hall PTR, 2003.9.

[6] KEN SCHWABER, MIKE BEEDLE. Agile Software Development with Scrum[M]. Upper Saddle River, New Jersey:Prentice Hall PTR, 2002.