ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.17, No.12, December 2006, pp.2450−2460
DOI: 10.1360/jos172450

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

一种风险驱动的迭代开发需求优先级排序方法*

黄 蒙[1,2+], 舒风笛[1], 李明树[1,3]

[1](中国科学院 软件研究所 互联网软件技术实验室,北京 100080)

[2](中国科学院 研究生院,北京 100049)

[3](计算机科学重点实验室(中国科学院 软件研究所),北京 100080)

A Risk-Driven Method for Prioritizing Requirements in Iteration Development

HUANG Meng[1,2+], SHU Feng-Di[1], LI Ming-Shu[1,3]

[1](Laboratory for Internet Software Technologies, Institute of Software, The Chinese Academy of Sciences, Beijing 100080, China)

[2](Graduate School, The Chinese Academy of Sciences, Beijing 100049, China)

[3](Key Laboratory for Computer Science (Institute of Software, The Chinese Academy of Sciences), Beijing 100080, China)

+ Corresponding author: Phn: +86-10-82620803 ext 804, E-mail: hm@itechs.iscas.ac.cn

Huang M, Shu FD, Li MS. A risk-driven method for prioritizing requirements in iteration development. *Journal of Software*, 2006,17(12):2450−2460. http://www.jos.org.cn/1000-9825/17/2450.htm

Abstract: Prioritizing requirements priority is the action that stakeholders assign the orders of requirements to be implemented. It is the basement of the iteration plan. Existent methods of prioritizing requirements are inadequate to support stakeholders' negotiation or the adjustment of requirements' priorities. These shortages always lead to a rigid iteration plan that is difficult to be adjusted to adapt changes of requirements and environments. In this paper, a risk-driven adaptive method is put forward for prioritizing requirements which combines adaptive planning and risk-driven methodologies. Requirements are prioritized adaptively, and risks are used as the foundation of priorities decisions. The negotiability and adjustability of requirements priorities can be enhanced by the method. The negotiable and adjustable requirements priorities improve the developers' capability of controlling requirements and reduce the faults of software project caused by requirements.

Key words: requirement priority; requirement negotiation; iteration development; adaptive planning; risk

摘 要: 需求优先级排序是指系统参与者为需求指定实现的优先次序,是迭代开发过程中开发者制定项目迭代计划的基础.现存的需求优先级排序方法对系统参与者之间的协商和调整优先级的支持能力不足,导致根据需求优先级所制定的迭代计划难以作出符合需求变更和环境改变的调整.提出一种风险驱动的需求优先级自适应排序方法.该方法将自适应计划方法学与风险驱动相结合,将风险作为排序决策的依据,以自适应的过程为迭代开发排序需求优先级.该方法能够改善需求优先级排序过程中系统参与者之间的协商和调整需求优先级的能力,增强在迭代开发中对需求的控制,降低因需求导致项目失败的可能性.

关键词: 需求优先级;需求协商;迭代开发;自适应计划;风险

* Supported by the National Natural Science Foundation of China under Grant Nos.60273026, 60573082 (国家自然科学基金); the National High-Tech Research and Development Plan of China under Grant No.2005AA113140 (国家高技术研究发展计划(863))

Received 2005-10-24; Accepted 2006-01-11

中图法分类号: TP311 文献标识码: A

需求变更和估算不准确是导致软件开发失败的两个最主要的原因[1].迭代开发是一种软件开发生命周期由多次迭代构成的软件构建方式.每个迭代是自包含的迷你项目(mini-project),每次迭代后产生一个部分完成的系统,最终迭代产生完整产品[2].迭代式的开发能够让开发者每次只估算和实现一次迭代所要完成的部分系统,从而改善开发者对系统的估算精度和控制需求变更的能力[3,4].

在迭代开发过程中,需求以分阶段、逐渐增加到系统上的方式完成.为了尽快满足用户的核心需要并保证开发过程受控,开发者需要对每次迭代所要实现的需求作出选择.需求优先级是系统参与者所指定的需求实现次序,需求优先级排序是迭代开发中所要解决的核心问题之一[3,5,6].系统参与者所选择的需求优先级决定了迭代方式能否获得在功能、非功能、费用、进度和质量方面都能让用户满意的软件产品[3,7,8].

需求优先级排序需要考虑多种因素,每个需求对于用户核心业务的价值、需求之间的依赖关系、开发团队可用资源、开发者和用户对系统目标和限制的理解程度、环境的演化等诸多因素都会影响到需求优先级,这使得在迭代开发过程中需求优先级排序变得非常复杂.需求优先级的相关研究[9−11]主要集中于研究评估需求价值,并在可用资源限制条件下确定优先级的方法,对系统开发过程中系统参与者对系统多个目标的理解深入和环境演化等因素均未作过多考虑.因此,根据这些方法所获得的需求优先级在迭代过程中难以协商和调整,使得开发者难以在需求和环境不断变化的情况下控制需求.具体来说,软件系统开发过程存在不同的参与者,这些参与者的目标各不相同,因此很难找到某个优先级方案,使得需求的实现次序可以恰好在最大程度上满足所有系统参与者的目标,这就需要为系统参与者提供多种优先级方案,使他们有机会考虑采用不同的优先级方案对软件开发目标的影响,进而对不同目标的满足程度加以折衷,使得软件开发取得令人满意的成果.此外,迭代过程是系统参与者对系统理解不断深入的过程,系统参与者对需求的业务价值、完成需求所需的工作量等会有越来越深的认识,这就要求在迭代过程中需要根据实际情况不断地调整需求优先级,以保证软件开发朝着正确的方向进行.而相关研究中提供的需求优先级排序方法很难同时满足以上要求.

为了解决以上问题,本文提出了一种风险驱动的迭代开发的需求优先级自适应排序方法.这种方法结合了风险驱动和自适应计划方法学(adaptive planning)[2],以与迭代过程进展同步的方式获得需求优先级.风险是不确定事件发生的可能性和带来的损失[12].本文方法中的风险分析为系统参与者协商和调整需求优先级提供了基准.这样,使得系统参与者能够在风险基础上进行有效的协商和合理的决策;自适应的排序过程能够让系统参与者在迭代活动中根据实际情况不断调整需求优先级,从而改善系统参与者在迭代过程中对需求的控制能力.

1 相关研究

一般来说,在迭代开发中排序需求优先级需要考虑以下问题:

(1) 在每次迭代的可用资源和需求依赖关系等限制条件内,将高收益、低成本的需求优先实现,从而提高用户的满意度和系统的可用性[10].其中,成本是指在系统中实现需求所需要的工作量;需求依赖关系是指需求所描述的业务过程之间的依赖关系,例如对于业务过程来说,业务事件 A 必须在业务事件 B 之前完成,那么描述 B 的需求依赖于描述 A 的需求;收益一般是指需求对用户业务的价值,通常使用需求重要性来表示.

(2) 建立开发者和用户的协商并调整需求优先级机制.在迭代过程中,随着开发者和用户对于系统理解的深入,会导致需求优先级发生变化[2,13].根据实际情况不断协商和调整早期定义的需求优先级,是保证开发出的软件产品持续满足用户需要的关键步骤[13−15].

在早期研究中,需求优先级排序主要是通过层次分析法(analytic hierarchy process,简称 AHP)[11]、质量功能部署(quality function deployment,简称 QFD)[16]等技术量化需求重要性,然后估算需求的成本,使用优化函数[9,10]寻找成本限制条件下的重要性之和最大的需求组,并对剩余需求迭代使用优化函数,直到所有的需求都被排序为止.在这些研究的基础上,一些研究还分析了多参与者的需求重要性冲突[17,18]和需求之间存在依赖关系的问题[11,19],并提供了相应的改进方案.另一方面,对迭代开发过程方面的研究一般都会涉及到需求优先级排序问题,

如螺旋迭代模型[20]和敏捷开发过程选择[21]中在成本、收益基础上使用风险作为驱动迭代中需求选择的根据.

这些研究关注于需求价值和开发团队可用资源之间的平衡,并提供技术来支持需求价值评估,平衡需求价值和可用资源,从而获得需求优先级的活动.但是,这些方法缺乏优先级的协商、决策和调整机制,通常只能给出单一的优先级方案.而软件开发的多目标特征决定了需求优先级一般不存在最优方案,因此,要为多个系统参与者提供多个方案供他们进行选择.这样,系统参与者可以根据不同方案对不同目标的满足程度进行协商,从而获得一致的意见,保证软件开发向着正确的目标进行.此外,这些研究提供的方法假设需求已经完全明确,需求优先级不需要进行大的调整.但是在迭代开发过程中很难第一次就把需求完全定义好[22],例如,不完美的人际沟通、复杂的商业过程等都会使得在项目开始的时候所确定的需求优先级不太准确.因此,需求优先级需要在迭代过程中不断加以调整[2,13].一些报告显示:没有从整体上考虑系统开发所指定的需求优先级,会导致优先级难以协商和调整,使得软件开发存在大量的返工甚至导致项目无法完成[7,8,23−25].因此,在需求优先级排序时,有必要为开发者和用户提供协商决策和调整优先级的途径.

近年来,敏捷开发方法学成为一种重要的迭代开发实践.文献[2]提出了两种适用于敏捷开发过程的基于自适应计划方法学的需求优先级排序方法.自适应计划方法学也称为波浪计划(rolling wave planning),其基本思路是:开发者和用户设定大粒度的时间里程碑,然后只为下一个迭代制定精确计划.自适应计划方法学为系统参与者提供了根据迭代成果不断改进后续开发活动的途径,为协商决策和调整需求优先级提供必要的过程支持.但是,该研究没有提供量化数据支持的需求优先级协商决策和调整技术,导致需求优先级排序过程中的协商决策和调整活动效率较低甚至难以进行.

随着COTS和开源组件在软件构造中应用范围的扩大,需求优先级排序成为基于COTS或开源组件开发过程中的重要问题.这是因为基于开源软件开发的特征是需要在需求和可用组件之间进行折衷;此外,对于开源组件来说,组件功能的演化速度较快,这在开发过程中又会反馈为需求变更的要求.这些特征导致协商决策和调整需求优先级是保证开发活动的正常进行的关键活动.在文献[26]中,我们给出了一种基于开源软件开发的需求和组件匹配过程.然而,该过程没有给出在需要分阶段构造的复杂系统开发情况下的需求优先级协商决策和调整的技术.

本文提出了一种风险驱动的迭代开发需求优先级自适应排序方法.该方法在使用成本收益分析的传统需求优先级排序技术的基础上,通过风险驱动的自适应排序过程来确定需求优先级.风险驱动是指将风险分析作为优先级协商决策和调整的基础.风险分析能够改进开发者对环境和组织能力的理解并增强在多个Stakeholder之间的一致性[21],在需求优先级排序过程中使用风险驱动,能够改善需求优先级自适应排序方法中缺乏协商和决策的支持数据所导致的优先级协商决策和调整活动效率较低等问题.

2 风险驱动的迭代开发需求优先级自适应排序方法

风险驱动的需求优先级自适应排序方法包括自适应排序的过程框架、多优先级方案生成算法和优先级方案的风险分析技术3个部分,下面分别加以介绍.

2.1 自适应排序过程框架

以不同次序实现需求会导致不同的开发结果,因此,需求优先级是在分析的基础上作出决策.从决策的过程来看,作出正确的决策首先要构造决策方案,然后对决策方案进行分析,从而确定出每个决策方案所导致的结果,最后作出决策.为了使决策过程能够以自适应的方式进行,从而保证系统参与者可以根据早期决策的结果来改进后期决策,我们在方法中设计了一种风险驱动的需求优先级自适应排序过程框架,如图1所示.该过程框架中包含3种活动,分别是为下一次迭代生成多个可行的需求组合;分析每个需求组合的风险;作出选择需求组合的决策.这个过程框架强调排序的目标是找到下一次迭代所应该实现的需求,而不是为所有需求和整个迭代过程制定需求实现次序.以这种方式所获得的需求优先级,可以通过度量每一次迭代后的成果来验证需求优先级决策的效果,从而获得后续迭代中优先级决策所需要的信息.

(1) 为下一次迭代生成多个可行的需求组合

为一下次迭代构造多个可行的需求组合是构造决策方案的活动.在此活动中,开发者需要搜集每个需求对用户业务的价值、需求的规模、需求之间的依赖关系以及一次迭代可完成的工作量等方面的信息,然后确定出多个对于下一次迭代可行的需求组合.可行的需求组合是指在工作量和依赖关系限制范围内收益成本较高的需求组合.可定义如下规则来寻找可行的需求组合:在满足可用工作量和依赖关系限制条件下,应优先选择那些价值之和较高的需求组合.提供多个可行的需求组合作为优先级方案来支持决策,可以帮助系统参与者发现采用不同的需求实现次序对系统开发所造成的影响.这些需求组合在经过风险分析之后将提供给系统参与者挑选.

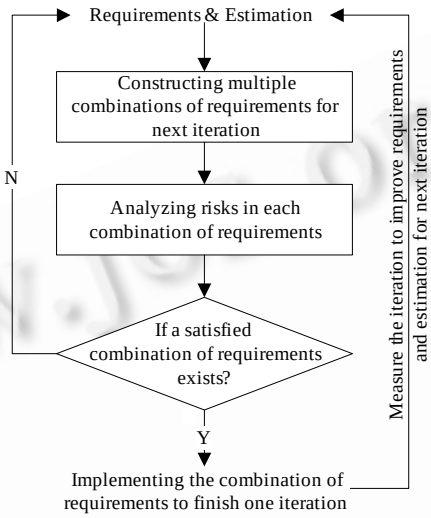

Fig.1 A process framework for risk-driven adaptive method for prioritizing requirements in iteration development

图 1 风险驱动的需求优先级自适应排序过程框架

(2) 分析每个需求组合的风险

分析每个需求组合的风险,是分析每个决策方案所可能导致后果的活动.分析需求组合风险包括识别风险和评估风险活动.识别风险的目标是识别出一个需求组合中存在的不确定事件.一个需求组合可能存在多种不确定事件,如易变的需求、难以设计和实现的需求、错误的需求规模估算和对团队生产率的不正确估计等;评估风险的目标是确定出这些不确定事件发生的可能性和带来的损失,进而得到一个需求组合的整体风险情况.不确定事件所带来的损失包括产品质量、成本和进度等方面.此外,在为下一次迭代选择了需求组合之后,风险分析的结果也用于执行迭代计划中的风险跟踪和控制,开发者根据所采用需求组合的风险情况跟踪风险,并在迭代活动中采取合适的措施控制风险,从而降低开发失败的可能性.

(3) 作出选择需求组合的决策

在此活动中,系统参与者根据每个需求组合的风险情况,经过协商,从多个需求组合中选择出下一次迭代所要实现的需求.历史数据支持的、基于风险的协商是实现该活动的关键.开发者根据每个需求组合中所包含的用户价值、所需要的成本和风险与过去的类似迭代进行对比,并依照当前项目的进展程度来作出决策.例如,在项目早期,应该尽可能地实现风险高且实现用户价值大的需求组合,这样可以尽快获得对系统的深入理解,从而保证软件开发活动向着正确的目标进行.而在靠近里程碑的迭代中,应该尽可能地实现风险小的需求组合,从而保证在里程碑发布可用的系统.在决策过程中,可以使用多指标决策的技术,系统参与者首先为进度、成本和产品质量 3 个风险的损失指标分配权重,然后将损失指标规范化,最后使用线性加权等多指标决策技术作出决策.

当系统参与者根据历史数据和经验难以选择出一个合适的需求组合时,说明项目需求和估算存在潜在的问题.此时,系统参与者需要返回早期的需求分析、成本估算等活动,重新考虑需求和成本估算中潜在的问题,如

是否应该继续精炼需求从而更准确地确定出用户的真正需要;判断估算是否过于乐观等等.然后,在修改需求和估算之后重新进行需求优先级排序.修改需求和估算活动可能涉及到调整项目的目标、范围、成本和进度计划等方面的内容.在这一过程中,可以使用如Win-Win模型等技术手段.这些活动在协商基础上进行,风险为其中的协商提供了必要的信息,从而促进系统参与者尽快达成一致.

在选定需求组合并执行完一次迭代后,系统参与者能够根据迭代成果验证优先级决策效果并改进后续迭代的决策.迭代过程和迭代成果的度量包括迭代实际成本、获得的用户价值等方面的信息,这些信息用于验证优先级排序过程中所获得的需求价值、成本估算、进度计划、风险等分析结果,通过与实际情况比对,系统参与者可以更好地理解系统的目标和限制,发现早期迭代对需求和估算的错误认识,从而改进后续迭代的开发活动.

尽管本文的方法强调为下一次迭代找出适于实现的需求组合,但该方法仍然可用于完整软件开发生命周期的需求优先级排序.例如,一个系统的开发计划使用3个里程碑点,那么,首先可以使用本文的方法找出里程碑1内要完成的需求组合,然后对剩余需求递归使用本文的方法,找出里程碑2内要完成的需求组合,最后再将剩余需求指定到里程碑3内完成.尽管如此,我们仍然认为这种全生命周期的需求优先级并不具有严格规范开发计划的意义,只能作为参考使用.这是由于随着系统参与者对系统目标的理解加深和环境的自然演化,开发过程必然存在着需求变更的情况,例如需求缺陷的更正、需求演化、更准确的需求成本估算等等.此时,早期指定的需求优先级次序可能失效,系统参与者需要重新指定需求优先级.一般来说,使用本文的方法只需要为大时间跨度的里程碑点和里程碑内的下一次迭代指定需求组合,保证开发过程以自适应的方式进行.

2.2 可行需求组合生成算法

在构造决策方案活动中,我们提供了一种为下一次迭代生成多个可行需求组合的算法.这种算法的目的是自动生成多个可行的需求组合,从而为构造决策方案提供支持.

为下一次迭代生成多个可行的需求组合的基本原则是在成本、依赖关系限制条件下找到多个高价值、低成本的需求组合.需求成本是指实现需求所需要的工作量,成本限制是指一次迭代可完成的工作量.约定如下:

n:假定未完成的需求集有 n 条.

L:表示期望找到的需求组合个数,$L\geq 1$.

$x_i=\{0,1\}$:表示在下一次迭代中是否实现需求 i,$x_i=1$ 表示在下一次迭代中实现需求 i;否则,$x_i=0$.

C_i:表示完成需求 i 所需要的工作量.

C^t:表示迭代 t 可完成的工作量.

$d_{ik}=\{0,1\}$:表示需求依赖关系,$d_{ik}=1$ 表示需求 r_i 依赖于需求 r_k;$d_{ik}=0$ 表示需求 r_i 不依赖于需求 r_k.指定$\forall i=k$,$d_{ik}=1$;

V_i:需求 i 对用户核心业务的价值.

其中,C_i 通过对需求规模的估算获得.C^t 根据项目的迭代时间长度以及团队生产率来确定.d_{ik} 在需求分析过程中获得.需求对用户业务的价值 V_i 由用户决定,价值可能包括客观的价值,如经济价值,也包括着主观的价值,如用户的满意程度等.对于排序需求优先级来说,只需要理解需求的相对价值[13].AHP 是建立不同对象之间相对价值的有效工具[27],可以使用该方法获得每个需求在0~1之间的一个值,值的大小代表相对商业价值的大小,所有需求值之和为1.

根据以上约定,将寻找可行需求组合的规则表示为以下目标函数和约束,分别如式(1)~式(3)所示.其中,$L\_\max$ 是指在所有解中 g 值最大的前 L 个;x_i 按照 V_i/C_i 降序排列,在原始需求集不能满足这一要求时,可以使用任何排序算法调整需求集中的需求次序以满足此要求.

$$L\_\max \qquad g=\sum_{i=1}^{n}V_i\times x_i \tag{1}$$

$$\text{s.t.} \qquad \sum_{i=1}^{n}C_i\times x_i\leq C^t \tag{2}$$

$$\forall x_i,x_k,\ x_i\times d_{ik}\le x_k,\ i,k\in\{1,\ldots,n\} \tag{3}$$

生成多优先级方案相当于对以上目标函数求解,由此,我们设计了算法 1.该算法是普通 0-1 背包问题回溯求解算法的改进.普通 0-1 背包问题不存在依赖关系限制和寻找前 L 个解的要求,无法直接应用到本文多需求优先级方案生成中.因此,我们在普通 0-1 背包问题回溯求解算法的部分可行解判定和完整可行解更新两处进行了改进,以处理依赖关系限制和寻找多个可行解.其中,V 和 C 表示 V_i 和 C_i 组成的向量;D 是 d_{ik} 组成的矩阵;X 是 x_i 不同取值的解向量.在算法中,S 表示包含着 L 个完整可行解的队列.完整可行解按照 g 值降序排列,其中 g 值最小的一个解向量记作 $X_{\min}$,其对应的值记作 $g_{\min}$.*Update*(S)用于更新 S,当算法所获得的一个完整解 X 对应的函数值 g 大于当前 S 队列中的 $X_{\min}$ 所对应的 $g_{\min}$,从 S 中删掉 $X_{\min}$,加入 X 并根据 g 值重新排序 S,可以使用任何排序算法来完成此任务.*backtrace* 函数用于在 X 中从 x_i 向 x_1 搜索找到第 1 个 x_k=1,将 $x_k\leftarrow 0$,$cu\leftarrow cu-C_k$,使算法能够寻找另一个完整的可行解,从而最终获得在所有可行的需求组合中 g 值最大的前 L 个.*ps* 函数为部分可行解的判定函数,即对于某个 x_l=1 或者 x_l=0,所得到的部分解能够满足约束条件.*ps* 函数中的 *bound* 用于进行边界检查,对于 x_l=0 时某个 x_1 到 x_{l-1} 的取值组合,从 x_{l+1} 到 x_n 即使在全部都等于 1 时,也不能产生出比当前 S 队列中最小的一个更好的解,那么,在应该直接舍弃该组合下寻求完整可行解的尝试,转而考虑其他 x_1 到 x_{l-1} 取值组合,从而加速求解速度.

算法 **1**. 多需求优先级方案生成算法.

```
procedure re_comb(V,C,C^t,X,D,n,g_min)
  for i←1 to n+1 do
    case
      :i=n+1 & X=0: return                          //exit //
      :i=n+1 & g_i>g_min: update(S); backtrace      //update S queue, and backtrace//
      :else: backtrace
    endcase
    case                                  // construct solution along x_i=1 or 0 or jump to other combination//
      :ps(1,i,D): x_i←1; cu←cu−C_i
      :ps(0,i,D): x_i←0
      :else: backtrace
      endcase
  repeat
end re_comb
procedure ps(x_i,i,D)
  if cu−C_i×x_i<0 then return (false) endif
    case                                  // dependencies restraints //
      : x_i=1: for j←1 to i−1 do
          if d_ij>x_j then return (false) endif
          repeat
      : x_i=0: for j←1 to i−1 do
          if x_j×d_ji>0 then return (false) endif
          repeat
    endcase
  for j←i+1 to n do bound←bound+C_j repeat
    if C^t−cu+bound<g_min then return (false) endif
  return (true)
end ps
```

算法 1 中,dependencies restraints 部分能够保证算法执行后所获得的完整解满足公式(3).

证明:按照公式(3),如果 x_1~x_{i-1} 的取值组合所构造出的部分可行解加入 $x_i\leftarrow 1$ 后仍为部分可行解,对于依赖关系限制来说,$\forall x_k,k\in\{1,\ldots,i-1\}$,应有 $d_{ik}\le x_k$.$x_i\leftarrow 0$ 时的情况类似.算法中提供了对 $x_i\leftarrow 1$ 和 $x_i\leftarrow 0$ 情况下部分解是否满足依赖关系限制的判定,保证了算法最终所获得的完整解满足公式(3). □

2.3 需求组合风险分析

分析需求组合风险的过程包括识别风险和评估风险活动:识别风险的目标是确定一个需求组合中存在的

不确定事件;评估风险的目标是确定出不确定事件发生的可能性和带来的损失.

风险分类表用于识别风险.开发者可以根据自己的需要设计自己的风险分类表.本文提供了一种适用于识别需求优先级风险的风险分类表,见表 1.

Table 1 A taxonomy for identifying risks during prioritizing requirements

表 **1** 用于识别需求优先级风险的风险分类表

Type	Risks	Notes
Requirements	Unstable requirements	Volatile requirements caused by environments
	Unclear requirements	Requirements are not enough clear and detailed for system goals and designing
	Difficult in designing and implementing	The requirements conflictions in architecture or hard to implement which are caused by technologies
Estimation	Wrong estimation of requirements size	Estimation of requirements size is wrong
	Wrong team productivity	Estimation of team productivity is wrong
People	Customers	Requirements not come from domain experts

评估风险包括确定出不确定事件发生的可能性和带来的损失,进而确定出每个不确定事件造成的风险暴露度(risk exposure,简称 RE),最后把多个不确定事件的风险暴露度组合在一起,得到需求组合的风险情况.

可以使用定性或定量的方法评估风险发生的可能性和损失.定量分析不确定性事件带来的损失需要较高的费用和时间投入,且有时难以搜集到足够的数据[28].因此,除了为一些关键系统的重大里程碑排序优先级以外,定性地分析风险发生的可能性和损失更适合于迭代过程.评估损失和可能性来源于根据开发者的经验以及历史数据的统计所得到的一些规则,如“不稳定的需求带来的质量损失为高”,这些规则通常是组织特定的.

本文在文献[27]研究的基础上给出了一种需求组合的风险暴露度定性评估方法.文献[27]方法使用低(low)、中(medium)和高(high)定性评估风险发生的可能性和损失,然后使用表 2 中的不可接受的(unacceptable)、关键的(critical)、重要的(significant)和一般的(minor)来描述风险暴露度.为了确定一个需求组合在产品质量/成本/进度方面的整体风险情况,并实现在定性方法基础上的定量分析,我们将定性的风险暴露度描述转化为风险暴露度得分(见表 2),并通过累计得出每个需求组合在产品质量、成本和进度方面的风险暴露度,这样为系统参与者比较不同需求组合的风险提供了直观的、可与历史数据比较的数据.使用式(4)来累计一个需求组合 k 在产品质量上的风险暴露度,累计进度和成本方面风险暴露度公式与其类似.

$$RE_{\text{产品质量}}(k)=\Sigma k \text{ 中产品质量方面风险暴露度得分} \tag{4}$$

最后,将每个组合的风险暴露度得分、所包含的用户价值、所需要的工作量等信息提供给开发者和用户,辅助他们进行协商决策.

Table 2 A qualitative method for evaluating risk exposure

表 **2** 风险暴露度定性评估方法

Description of risk exposure (Score of risk exposure)		Possibility		
		Low	Medium	High
Lost	Low	Minor (1)	Significant (2)	Critical (3)
	Medium	Significant (2)	Critical (3)	Unacceptable (4)
	High	Critical (3)	Unacceptable (4)	Unacceptable (4)

3 应用研究

本文提出的风险驱动的需求优先级自适应排序方法可与 eXtreme Programming(XP)等敏捷方法学结合使用.我们通过应用研究对该方法进行了验证,结果显示:使用该方法在 XP 实践中更容易进行需求协商,并改善了开发者的估算及控制项目进展的能力.

3.1 应用背景

我们针对一个项目,安排了两个人员组成结构、能力以及经验近似的项目组进行 XP 实践.研究项目的背景特征见表 3.在参与应用的两个项目组中,一组使用标准的 XP 实践(以下简称为对照组 A);另一组在 XP 实践中

使用本文所提出的风险驱动的需求优先级自适应排序方法(以下简称为方法组 B).由于风险驱动的需求优先级自适应排序方法的使用,方法组 B 的 XP 实践软件过程与一般的 XP 过程不同,如图 2 所示.

Table 3 Characteristics of the project

表 3 项目特征

Characteristics of the project	Description
Team size	6 developers
Type of end product	Web applications
Developers' experience in XP/agile	1 experienced, 5 novice
Developers' experience in end product	6 experienced
Developers' experience in coding	6 experienced
Development tools	JBuilder 9, JDK1.4, Tomcat4, Mysql; Bugrat, Winrunner; QMP2.5, Firefly2.5

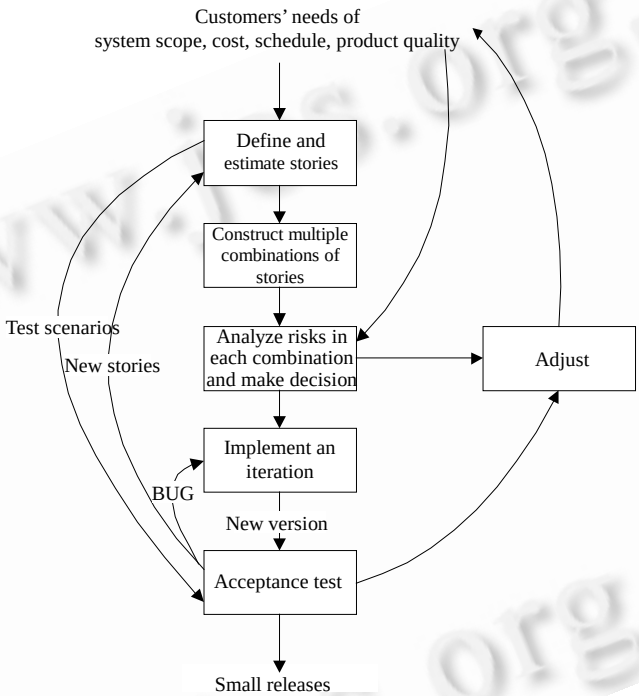

Fig.2 An XP process in which the risk-driven adaptive method for prioritizing requirements is used

图 2 加入风险驱动的优先级自适应排序方法的 XP 过程

3.2 项目数据搜集

我们搜集了客观和主观两类项目数据.客观数据来自于组织的项目跟踪、缺陷跟踪、源代码统计和测试工具,如图 3 所示,并见表 4 的说明.

使用调查问卷表来搜集开发者对风险驱动的优先级自适应排序方法的主观反馈意见.问卷表中包含了一些具有标准单项选择的问题,选项包括“很好”、“好”、“一般”和“无效”.除此之外,我们也给出了一些开放问题,以用于搜集开发者的观点.搜集到的单选问题反馈意见汇总结果见表 5.

3.3 数据分析与结论

通过对比项目数据项可以发现:两个团队在编码的生产率和最终产品的规模方面类似,但在进度和工作量方面存在较大差别.其中,方法组 B 比对照组 A 提前两周交付了完整系统,工作量减少了 21%((2536.5–1991.5)/2536.5).此外,在交付系统之后,用户对系统的满意程度也有所不同:方法组 B 的实践中保留了一个用户不满意的故事,留待未来系统升级时实现;而对照组 A 实践中则有 3 个用户不满意的故事被保留下来.

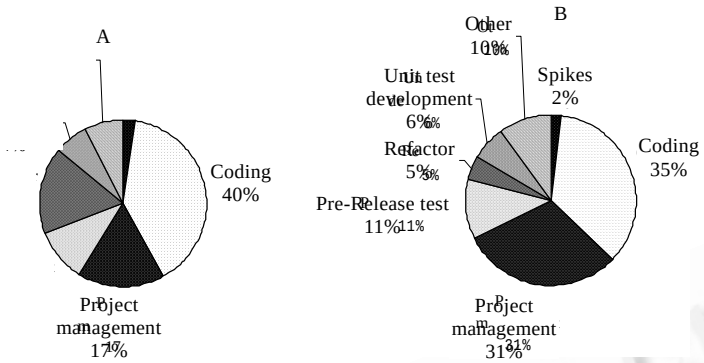

Fig.3 Effort of tasks

图 3 工作量分布

Table 4 Project data

表 4 项目数据

No.	Data*		Iteration No.					Total
			1	2	3	4	5	
1	Calendar (weeks)	A	3	2	2	2	2	11
		B	3	3	2	1	/	9
2	Efforts (personal hour)	A	609	638	423.5	399	467	2 536.5
		B	613	703	476	199.5	/	1 991.5
3	Contributions to end product (KLOC)	A	13.22	11.99	15.46	20.71	21.6	82.98
		B	21.89	26.5	19.52	8.02	/	76.9
4	Team productivity (loc/hour)	A	36.1	38.0	39.7	40.9	40.9	39.1
		B	35.7	37.7	41.0	40.2	/	38.6
5	The Numbers of requirements changes after iteration	A	5	4	6	5	3**	23
		B	7	4	1	1**	/	13
6	The numbers of stories be implemented	A	10	10	8	2	4	34
		B	11	9	10	6	/	36
7	The numbers of Bug after acceptance test	A	0	0	0	0	0	0
		B	0	1	0	0	/	1

* Data item:

A—Data from the team that apply common XP practice in the project

B—Data from the team that apply the risk-driven adaptive method for prioritizing requirement in the project

** Preserved for future

Table 5 Feedback from developers

表 5 开发者的反馈

The risk-driven adaptive method for prioritizing requirements in iteration development are useful in:	Feedback of developers (Numbers of persons agreement)			
	Very good	Good	Fair	Poor
Speeding up release planning	3	3	0	0
Balancing project scope and available resources	4	2	0	0
Controlling project progress	2	3	1	0

从图 4 的需求变更率(变更的故事数/最终交付的故事数)对比可以发现:在对照组 A 的后期迭代中,需求发生的变更数量很多,这些需求的变化导致了很多不必要的返工,从表 4 中的“每次迭代为最终产品贡献的代码行(contribution to end product)”也可以得到这个结论.而在方法组 B 中,需求变更的数目下降得很快,这意味着开发者和客户很快对项目的目标、范围和限制达成了一致,从而减少了早期迭代的无效代码所导致的后期返工.这可以解释为在生产率近似的情况下,一般的 XP 实践需要更多的成本和遗留下更多的、未正确实现的需求.开发者的主观意见也验证了这一结论,他们认为:风险驱动的需求优先级自适应排序方法对于项目范围和可用资源的协商是有效的,如图 4 所示.

我们通过在项目中应用风险驱动的需求优先级排序方法得到如下的初步结论:

(1) 在需求优先级排序时,通过风险识别和评估,可以帮助系统参与者改进他们对系统目标和自身能力的理解.在风险评估的基础上,系统参与者可以在系统目标和资源之间作出折衷,保证开发过程向正确的方向

进行.

(2) 方法改进了系统参与者之间协商决策的效率.开发者通过为下一次迭代构造出多个可行的需求组合并揭示出它们的风险情况,能够有效地支持系统参与者的需求优先级协商和决策.

(3) 可通过提供相应的工具来支持、改进该方法的具体应用.例如,工作量分布的统计数据(如图 3 所示)显示:用于项目管理(其中包含着协商和做出决策的活动)所占的工作量比例要大于一般的 XP 实践,因此,相应的协商和决策支持工具能够改进该方法的实际应用;此外,一个需求可能出现在多个需求组合中,若能通过相应工具支持多个需求组合中的风险数据的复用,可以减少在分析风险和制定决策时进行的一些不必要的重复性工作.

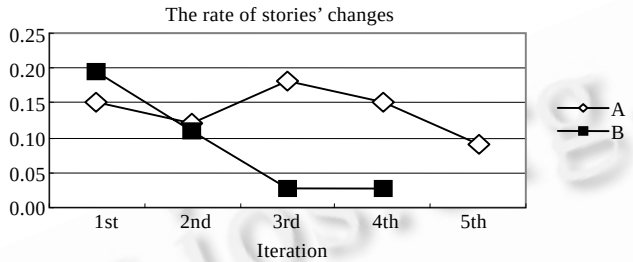

Fig.4 Comparison of story change rate

图 4 需求变更率比较

4 结 论

需求优先级是制定迭代开发计划的基础,需求优先级排序是迭代开发中开发者所需要考虑的核心问题之一.根据实际情况不断协商和调整需求优先级,决定了迭代开发能否获得在功能、非功能、费用进度和质量方面都能让用户满意的软件产品.这首先要求为系统参与者提供多种需求优先级方案,以及采用不同的方案对系统开发整体产生的影响等方面的信息,以供他们作出决策;此外,还需要在每次迭代完成后,开发者和用户能够根据软件开发过程中出现的情况对需求优先级作出调整,从而保证所制定的迭代计划符合实际情况,并保证开发活动向着正确的目标进行.目前的需求优先级排序方法所得到的需求优先级难以协商和调整,因此会给迭代开发过程带来很多问题.如只提供单一优先级方案,难以支持系统参与者从多个角度分析优先级决策对系统开发整体所造成的影响.协商决策过程中缺乏支持数据,使得系统参与者对需求优先级难以达成一致等等.

本文提出了一种适用于迭代开发的风险驱动的需求优先级自适应排序方法.这种方法结合了风险驱动和自适应计划方法学,以自适应的方式进行需求优先级排序并将风险作为优先级决策的根据,能够改善开发者在需求优先级排序过程中协商和调整需求优先级的能力,从而使开发者获得更好的需求控制能力,降低因需求所导致的软件开发失败的可能性.本文提出的方法可以应用于不同的迭代开发模式,我们将这种方法应用于 XP 的迭代过程,结果显示,该方法能够改进 XP 的开发效率.

在未来的工作中,我们将研究为不同的迭代开发方法学提供加入风险驱动的需求优先级自适应排序方法后的开发过程指导.同时,根据方法设计出辅助开发者进行数据搜集和分析的工具,以减少他们在需求优先级排序过程中的工作量.

References:

[1] Glass RL. Facts and Fallacies of Software Engineering. Boston: Addison Wesley Professional, 2002. 67−70.

[2] Larman C. Agile & Iterative Development: A Management's Guide. New York: Pearson Education, 2004. 23−24.

[3] Paetsch F, Eberlein A, Maurer F. Requirements engineering and agile software development. In: Proc. of the 12th IEEE Int'l Workshops on Enabling Technologies (WETICE 2003). Los Alamitos: IEEE Computer Society Press, 2003. 308−313.

[4] Sommerville I. Software Engineering. 6th ed., Boston: Addison Wesley, 2000. 121−124.

[5] Regnell B, Karlsson L, Höst M. An analytical model for requirements selection quality evaluation in product software development. In: Proc. of the 11th IEEE Int'l Conf. on Requirements Engineering (RE 2003). Los Alamitos: IEEE Computer Society Press, 2003. 254−263.

[6] Carlshamre P. Release planning in market-driven software product development: Provoking an understanding. Requirement Engineering, 2002,7(3):139−151.

[7] Jackson A, Tsang SL, Gray A, Driver C, Clarke S. Behind the rules: XP experiences. In: Proc. of the Agile Development Conf. (ADC 2004). Los Alamitos: IEEE Computer Society Press, 2004. 87−94.
[8] Rand C, Eckfeldt B. Aligning strategic planning with agile development: Extending agile thinking to business improvement. In: Proc. of the Agile Development Conf. (ADC 2004). Los Alamitos: IEEE Computer Society Press, 2004. 78−82.
[9] Jung HW. Optimizing value and cost in requirements analysis. IEEE Software, 1998,15(4):74−78.
[10] Karlsson J. Software requirements prioritizing. In: Proc. of the 2nd Int'l Conf. on Requirements Engineering (RE'96). Los Alamitos: IEEE Computer Society Press, 1996. 110−116.
[11] Ruhe G, Greer D. Quantitative studies in software release planning under risk and resource constraints. In: Proc. of the 2003 Int'l Symp. on Empirical Software Engineering (ISESE 2003). Los Alamitos: IEEE Computer Society Press, 2003. 262−271.
[12] Boehm B, Ross R. Theory-W software project management principles and examples. IEEE Trans. on Software Engineering, 1989, 15(7):902−916.
[13] Beck K, Fowler M. Planning Extreme Programming. Boston: Addison-Wesley Professional, 2000. 77−78.
[14] Greer D, Ruhe G. Software release planning: An evolutionary and iterative approach. Information & Software Technology, 2004, 46(4):243−253.
[15] Lehman MM, Ramil JF. Rules and tools for software evolution planning and management. Annals of Software Engineering, 2001, 11(1):15−44.
[16] Tran T, Sherif JS. Quality function deployment (QFD): An effective technique for requirements acquisition and reuse. In: Proc. of the 2nd IEEE Software Engineering Standards Symp. Los Alamitos: IEEE Computer Society Press, 1995. 191−200.
[17] Moisiadis F. The fundamentals of prioritising requirements. In: Proc. of the Systems Engineering/Test and Evaluation Conf. (SETE 2002). 2002 (Web). http://www.seecforum.unisa.edu.au/Sete2002/ProceedingsDocs/
[18] In HP, Olson D, Rodgers T. Multi-Criteria preference analysis for systematic requirements negotiation. In: Proc. of the IEEE Int'l Computer Software and Applications Conf. (COMPSAC 2002). Los Alamitos: IEEE Computer Society Press, 2002. 887−892.
[19] Carlshamre P, Sandahl K, Lindvall M, Regnell B, Dag JN. An industrial survey of requirements interdependencies in software product release planning. In: Proc. of the 9th IEEE Int'l Conf. on Requirements Engineering (RE 2001). Los Alamitos: IEEE Computer Society Press, 2001. 84−93.
[20] Boehm B. A spiral model of software development and enhancement. IEEE Computer, 1988,21(5):61−72.
[21] Boehm B, Turner R. Using risk to balance agile and plan-driven methods. IEEE Computer, 2003,36(6):57−66.
[22] Kulak D, Guiney E. Use Cases: Requirements in Context. Boston: Addison-Wesley, 2000. 12−14.
[23] Nawrocki J, Jasinski M, Walter B, Wojciechowski A. Extreme programming modified: Embrace requirements engineering practices. In: Proc. of the 10th Int'l IEEE Conf. on Requirements Engineering. Los Alamitos: IEEE Computer Society Press, 2002. 303−310.
[24] Lippert M, Becker-Pechau P, Breitling H, Koch J, Kornstädt A, Roock S, Schmolitzky A, Wolf H, Züllighoven H. Developing complex projects using XP with extensions. IEEE Computer, 2003,36(6):67−73.
[25] Cao L, Mohan K, Xu P, Ramesh B. How extreme does extreme programming have to be? Adapting XP practices to large-scale projects. In: Proc. of the 37th Hawaii Int'l Conf. on System Sciences (HICSS 2004). Washington: IEEE Computer Press, 2004. 34−41.
[26] Huang M, Yang LG, Yang Y. A development process for building OSS-based applications. In: Li M, Boehm B, Osterweil LJ, eds. Proc. of the SPW 2005. LNCS 3840, Berlin, Heidelberg: Springer-Verlag, 2005. 122−135.
[27] Karlsson J, Wohlin C, Regnell B. An evaluation of methods for prioritizing software requirements. Information and Software Technology, 1998,39(14-15):938−947.
[28] Greer D, Bustard DW. Towards an evolutionary software delivery strategy based on soft systems and risk analysis. In: IEEE Symp. on Engineering of Computer Based Systems. Washington: IEEE Computer Press, 1996. 126−133.

黄蒙(1973－),男,黑龙江肇东人,博士生,主要研究领域为智能软件工程.

李明树(1966－),男,博士,研究员,博士生导师,CCF 高级会员,主要研究领域为智能软件工程,实时系统.

舒风笛(1976－),女,博士,助理研究员,主要研究领域为需求工程,软件过程,软件估算.