上海理工大学学报
第36卷 第6期 J. University of Shanghai for Science and Technology Vol.36 No.6 2014

文章编号:1007-6735(2014)06-0578-07 DOI:10.13255/j.cnki.jusst.2014.06.013

基于快速应用开发的功能点增量迭代模型

夏骄雄[1]， 刘 政[2,3]， 刘绪彬[4]， 宋 阳[3]， 袁佳锦[4]

(1.上海市教育委员会 信息中心,上海 200003; 2.百度(中国)有限公司,上海 201203;
3.上海大学 计算机工程与科学学院,上海 200444; 4.上海理工大学 光电信息与计算机工程学院,上海 200093)

摘要: 针对快速应用开发模型、增量模型和敏捷软件开发方法等普遍存在的因控制不当而蜕变为边做边改模型等问题,一种使用敏捷软件开发方法的基于快速应用开发的功能点增量迭代模型正式提出.它通过切分流程、生成增量序列、使用项目进度池和缺陷(bug)状态池等手段,重新构筑了功能点的迭代过程.项目进度池负责跟踪软件系统开发的进度,bug状态池则负责装载用于后期质量评估的代码bug反馈,从而提升软件项目的开发效率.最后,通过实例应用表明功能点增量迭代模型在实际应用中具有较好的效果.

关键词: 软件工程; 开发模型; 功能点增量迭代
中图分类号: TP 311.52 **文献标志码**: A

Incremental Story Iteration Model Based on Rapid Application Development

XIA Jiao-xiong[1], LIU Zheng[2,3], LIU Xu-bin[4], SONG Yang[3], YUAN Jia-jin[4]

(1. *Information Centre, Shanghai Municipal Education Commission, Shanghai 200003, China*;
2. *Baidu (China) Co. Ltd., Shanghai 201203, China*; 3. *School of Computer Engineering and Science, Shanghai University, Shanghai 200444, China*; 4. *School of Optical-Electrical and Computer Engineering, University of Shanghai for Science and Technology, Shanghai 200093, China*)

Abstract: In view of the characteristic of changing qualitatively into build-and-fix model by the bad control, which is a universally existing problem about these software development models, such as rapid application development (RAD) model, incremental model, and agile software development, a new kind of model by the name of incremental story iteration model on rapid application development (ISI-RAD) was presented based on agile software development. With the process segmentation and the incremental sequence generation, the project schedule pool and the bug state pool were used to construct the story iteration. The project schedule pool can be responsible for tracking the software system development schedule, and the bug state pool can be responsible for loading the code bug feedback to evaluate the later quality, and then the efficiency of software development can be improved. Good results were achieved in practical application of ISI-RAD to an example.

收稿日期: 2013-10-01
基金项目: 国家自然科学基金资助项目(L61303097); 上海市重点学科建设资助项目(J50103)
第一作者: 夏骄雄(1973-),男,副研究员.研究方向:数据挖掘、软件辅助设计.E-mail:jshardrom@shmec.gov.cn

Key words: *software engineering*; *development model*; *incremental story iteration*

当前,软件项目的需求变化越来越快,功能越来越多,架构越来越复杂,软件开发模型的演变和发展也越来越纷繁复杂[1-4].传统的瀑布模型、螺旋模型、快速原型等早已不能适应越来越复杂和不断变化的需求及市场环境[5-6].因此,计算机领域的专家学者、IT 领域从业人员都进行了大量工作,提出了一些让软件开发团队具有快速交付、即时响应的价值观和原则.至 20 世纪 90 年代末,极限编程、自适应软件开发、动态系统开发等一系列敏捷软件开发(agile software development)方法相继被提出[7-8].

敏捷软件开发是一种以人为核心、迭代、循序渐进的开发软件方法,目标是提高开发效率和响应能力.但是在具体应用阶段,敏捷开发对于开发人员而言,由于用户需求的不确定性,往往就演变成边做边改模式,违背了执行敏捷开发的基本原则[9-10].因此,本文针对这一情况,根据多种软件开发模型,结合客观实际,提出了一套具体实施敏捷开发的步骤模型,即基于快速应用开发的功能点增量迭代模型(incremental story iteration model on rapid application development,ISI-RAD).

1　常用软件开发模型

软件开发模型是指软件开发全部过程、活动和任务的结构框架,它明确规定了要完成的主要活动和任务[1].与本文提出的基于快速应用开发的功能点增量迭代模型 ISI-RAD 有关的软件开发模型或方法有:边做边改模型[11]、快速应用开发 RAD 模型[12-13]、增量模型[14]和敏捷软件开发方法[7-10,15].

1.1　边做边改模型

边做边改模型是一种类似于作坊式的软件开发方法,既没有规格说明书,也没有设计文档.开发过程中,开发人员拿到项目立即根据需求编写程序,调试通过后生成软件的第一个版本.在提供给用户使用后,如果程序出现错误,或者用户提出新的要求,开发人员重新修改代码,直到用户满意为止.因此,开发人员需要根据用户不断变化的需求进行一次又一次的修改.对于需求简单、规模较小的项目,边做边改模型非常合适、非常灵活、需求响应快.

在需求复杂、结构庞大的项目中,由于缺少规划和设计环节,忽略需求环节,软件结构随着修改次数的不断增加而变得复杂且混乱,给软件开发带来较大风险.且由于没有考虑测试和程序的可维护性,没有任何开发设计文档,软件的维护十分困难.

1.2　RAD 模型

RAD 模型是一种增量型的软件开发过程模型,具有极短的开发周期.RAD 模型是传统生命周期模型的一类“高速”变种,通过大量使用可复用的构件,采用基于构件的建造方法获得快速开发的效果.在用户需求能够被较好理解、软件边界固定和明确的情况下,RAD 模型能够在极短的时间内创造出功能完善的系统.其基本流程包括业务建模、数据建模、过程建模、应用生成、测试及反复五大部分.

对于大型应用系统的开发而言,RAD 模型需要足够的人力资源支撑,开发人员和用户必须在短时间内做好整个系统的需求分析,并确保交互沟通的实时性和畅通性,否则将严重影响 RAD 模型的使用效率.对于不能合理划分模块化的应用系统,或者性能需求较高且构件接口需要调整的应用系统,RAD 模型并不具备很强的适应性.RAD 模型不适合于大范围使用没有应用经验的技术,否则任何一个小问题都会影响整个开发过程的进度.

1.3　增量模型

增量模型是融合了瀑布模型的基本成分和快速原型模型的迭代特征,采用时间进展的交错线性序列,使每一个线性序列都能产生软件产品一个可发布的“增量”.通常情况下,使用增量模型产生的第一个增量就是核心产品,实现了软件最基本的需求.增量模型引进了增量包的概念,只需要某个需求增量包产生,就可以进行相应的开发.用户对每一个增量的使用和评估都将作为下一个增量发布的新特征和功能,这个过程在每一个增量发布后不断重复,直到最终产生完善的产品.

增量模型的缺陷在于:一是要求软件本身具备开放式的体系结构,以便加入的增量不破坏已经构造完整的系统整体;二是增量模型的灵活性能够适应开发过程中需求的变化,但是仍存在退化为边做边改模型的可能性,使软件开发过程的控制失去整体性;三是如果增量包之间存在相交的情况,而且未加以正确处理,将导致系统分析的重新实施.

1.4　敏捷软件开发方法

敏捷软件开发又称敏捷开发,是 20 世纪 90 年

代开始逐渐引起广泛关注的新型软件开发方法.相对于“非敏捷”而言,敏捷开发更强调软件开发中人的作用,包括:开发人员团队与业务专家之间的紧密协作、面对面的沟通、频繁交付新的软件版本、紧凑而自我组织型的团队、能够较好适应需求变化的代码编写和团队组织方法等.

正是由于敏捷开发过于强调人的作用,极易导致开发过程与工具的使用相对减少、完整开发过程的文档留存相对薄弱、针对性的人与人沟通效率趋于瓶颈、软件维护的后期成本趋于扩大等问题,从而让敏捷开发模型失去约束,丧失活力,蜕化为边做边改模型.

2 ISI-RAD模型的基本结构

由于常用的软件开发模型或多或少都会存在一些问题,在实际应用中往往需要结合几个模型一起来完成软件的开发工作.所以,本文借鉴以前的研究成果[16],提出了一套具体实施敏捷开发的步骤模型,即ISI-RAD模型.

ISI-RAD模型是利用RAD模型、增量模型及敏捷开发方法的综合优势,结合其它软件开发模型的优点,摒弃缺点,以实际经验和实践总结出来的一套改良型的敏捷开发模型,其基本结构如图1所示.ISI-RAD模型主要由RAD部分、增量部分和功能点(story)迭代部分组成.

2.1 RAD部分

ISI-RAD模型选取RAD模型的业务建模、数据建模、过程建模等流程对软件系统进行切分,把其切分成若干个独立的模块[17].这样,软件系统的基本结构趋于简单化,方便管理,开发人员团队可以分组进行任务的完成,并可以自行安排任务的完成.

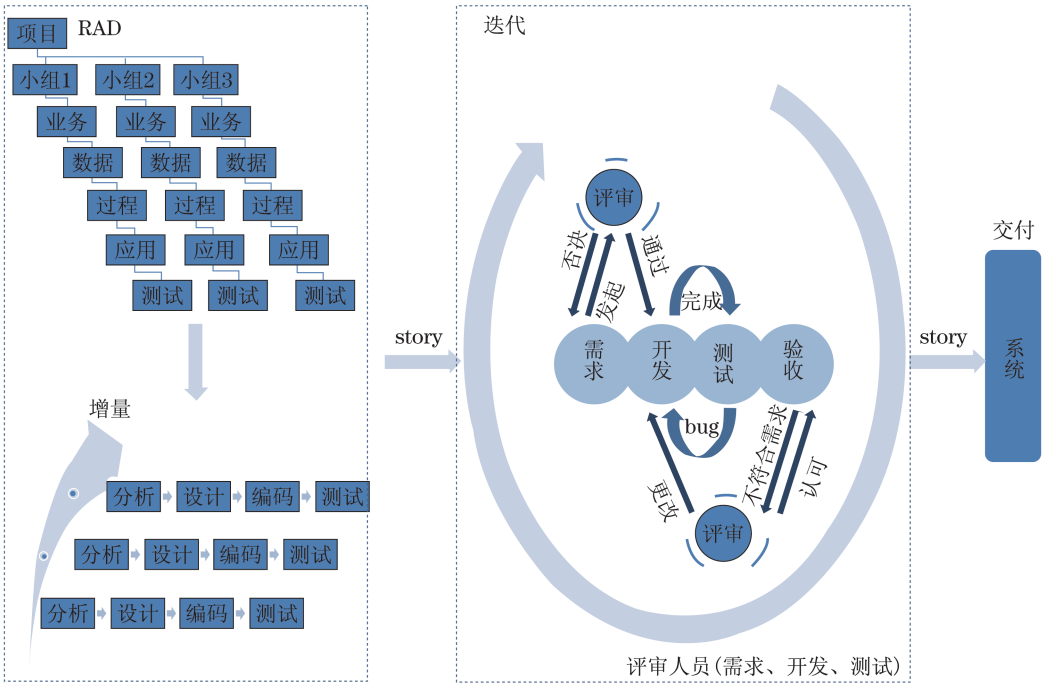

图1 ISI-RAD模型基本结构

Fig.1 Basic structure of ISI-RAD model

2.2 增量部分

ISI-RAD模型通过使用增量模型,生成一个增量周期序列,其中第一个增量定义为软件系统的核心功能,实现最基本的功能需求.然后,把各个模块的第一个增量提取出来进行需求分析,分析后进行story拆分并进行原型设计.最后,用户对当前增量的使用意见都作为下一个增量发布的新特征和功能.

增量部分的过程是在每一个增量发布后不断重复,直到产生最终的完善产品.过程中对story拆分并进行原型设计的目的在于把重要的story优先开发出来,利用成熟的增量模型进行风险控制,加大对ISI-RAD模型实施的依据,使得软件系统的开发是宏观有序的,而不是随意的[18].

2.3 story迭代部分

ISI-RAD模型通过使用敏捷软件开发方法,把拆分好的story存放在项目进度池和缺陷(bug)状态池中,如图2所示.假设有功能点story 1~15进行迭代,从而取得所需要的迭代结果,交付用户不同可用的版本[15].

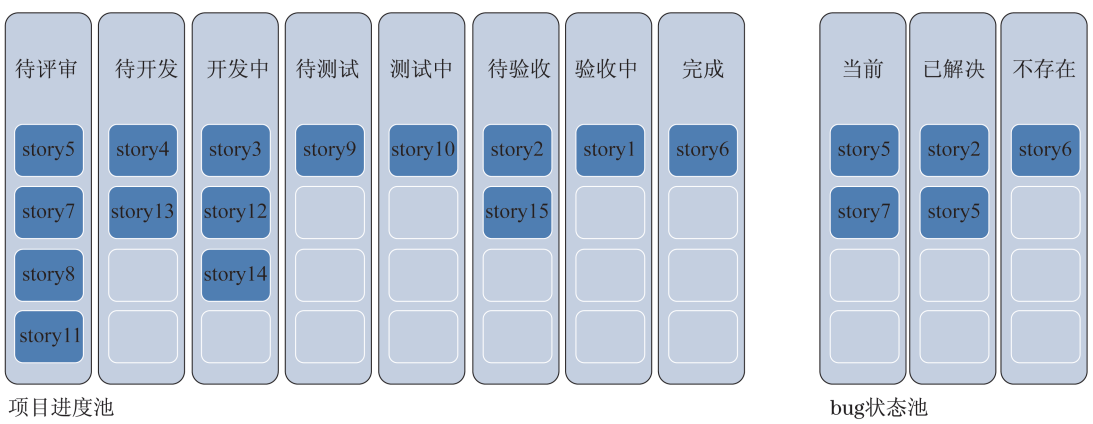

图 2　ISI-RAD 模型中的项目进度池和 bug 状态池

Fig. 2　Project schedule pool and the bug state pool of ISI-RAD model

项目进度池是存放 story 的地方，包含待评审、待开发、开发中、待测试、测试中、待验收、验收中、完成共 8 个状态.

Bug 状态池是存放 story 相关 bug 的地方，包含当前、已解决、不存在(误报或者无法复现)共 3 个状态.

3　ISI-RAD 模型中 story 迭代过程

ISI-RAD 模型中 story 迭代过程主要由项目进度池迭代过程和 bug 状态池迭代过程组成.

3.1　项目进度池迭代过程

项目进度池的功能是装载软件系统开发的整个过程，负责跟踪软件系统开发的进度情况，便于形成软件系统开发的进度报告，其迭代过程如图 3 所示.

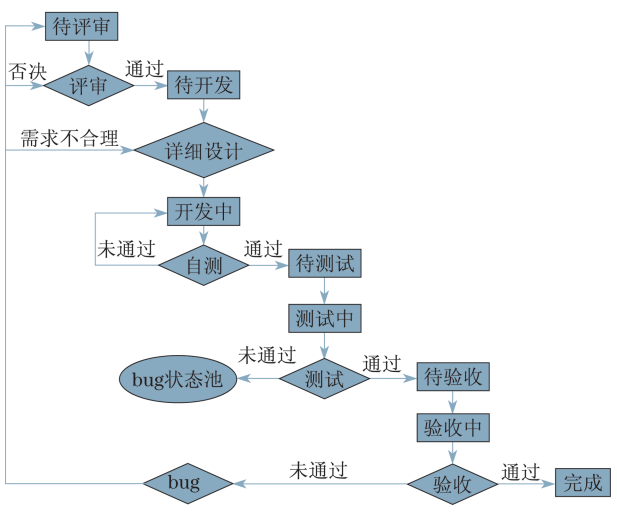

图 3　项目进度池迭代过程

Fig. 3　Iterative procedure of the project schedule pool

项目进度池迭代步骤如下：

步骤 1　把拆分好的 story 放入项目进度池，设置状态为待评审状态.

放入的方式有两种：

a. 先对整个软件系统开发进行需求分析设计，然后再拆分 story，对 story 进行评级，根据级别把优先级高的拿到第 1 期去做.

b. 只针对优先级高的 story 进行需求分析设计，然后拆分，直接纳入第 1 期去做.

方式 a 的优点在于整个软件系统开发过程有统一规划，各节点的拼装都是在规划蓝图下完成的；其缺点是类似于瀑布模型，需要等到整个需求分析完成后才能进行下一步，虽然拆分后可以实施敏捷开发，但是在这一步造成的“非敏捷”状态，将引发长期的阻塞.

方式 b 的优点是相对于方式 a 更灵活，只需要大致知道软件系统的结构规划即可，针对较高优先级的需求，优先分析设计及拆分，尽早投入开发；其缺点是以后模块集成时，可能存在设计缺陷，不能满足后续的需求，需要更改第 1 期的代码.

步骤 2　需求人员跟开发人员、测试人员一起，对拆分好的 story 进行评审.

评审内容包括业务上是否合理、技术上是否可行，story 是否过大，对 story 的测试是否确认等. 这一步几乎是所有角色都需要参与的，把第 1 期的 story 给相关人员进行评审，评审合格后纳入待开发状态，不合格的需要退回上一步修改.

步骤 3　对处于待开发状态的 story，参照需求文档进行详细设计.

详细设计的关键在于了解相应 story 在整个软件系统开发中的作用. 详细设计完成以后，指定相应的开发责任人，由该责任人对该 story 负责，参照 story 优先级自行决定何时将 story 的状态更改为开

发中状态.假如由于需求不合理,评审的时候没有充分考虑,导致后期 story 详细设计时无法实现的问题,则与相关责任人沟通.若确实需要进行重新需求分析,则将该 story 的状态更改为待评审状态,使之重新评审.

步骤 4 对处于开发中状态的 story,测试人员给出针对该 story 的测试用例.测试用例的提出,有助于开发人员在开发完成后进行自测,提高代码质量,减少沟通成本.当开发人员自测通过后,将该 story 的状态更改为待测试状态,若不通过则继续开发.

步骤 5 测试人员搭建测试环境.

测试人员给出针对该 story 测试用例之后,需要搭建相应的测试环境,然后参照 story 的优先级,将该 story 的状态更改为测试中状态.

步骤 6 测试人员进行测试实施.

测试人员按照该 story 的测试用例,在相应的测试环境对该 story 进行测试.测试通过后将该 story 加入整个软件系统,并进行集成测试.集成测试通过后,将该 story 的状态更改为待验收状态;若测试未通过,则生成 1 个 bug,并给予相应的"高、中、低"3 个状态.

Bug 若标识为"高"状态,则需要中断当前 story 的开发,即刻修复该 bug;若标识为"中"状态,则需要等待当前 story 开发完成后,进入待测试状态,再修复该 bug;若标识为"低"状态,则按照开发人员自己的进度安排,自行决定何时进行修复,但必须在软件系统交付之前给予解决.

步骤 7 需求人员确定验收的具体安排.

需求人员根据任务安排,并参照 story 的优先级,确定验收的先后顺序,同时将相应 story 的状态更改为验收中状态.

步骤 8 需求人员和用户共同实施验收.

需求人员和用户按照需求进行验收.验收通过后,将该 story 的状态更改为完成状态;若验收不通过,通知开发人员和测试人员召开站会(dispatch meeting,指以站立方式召开的、约耗时 10～15 min 的会议形式),讨论不通过的理由[7].如果确实是需求不完善或者变更造成的不通过,则必须进行 story 重新审核,再走一次流程即可.

3.2 Bug 状态池迭代过程

Bug 状态池用于装载软件系统开发过程中代码 bug 的反馈,便于后期的质量评估.

Bug 状态池的迭代过程如图 4 所示,其过程相对容易,只是记录 bug 的当前状态,需要关联项目进度池里相应的 story.如果开发人员提交测试后,测试人员发现存在 bug,那么只需要将 bug 放入 bug 状态池中,开发人员则根据优先级,决定修复的先后次序.

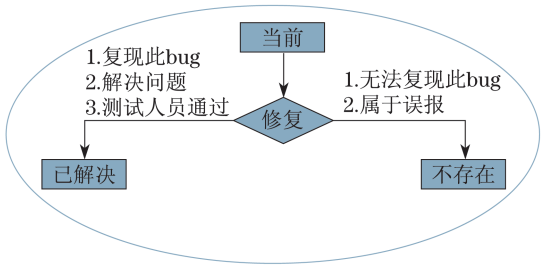

图 4 bug 状态池迭代过程

Fig.4 Iterative procedure of the bug state pool

4 ISI-RAD 模型的优势

由于 ISI-RAD 模型综合了常用软件开发模型的长处来完成软件的开发工作,因此 ISI-RAD 模型的优势就集中体现在工作精细化和周期快捷化等领域.

例如,在开发某公司运营系统的业绩稽核功能时,根据用户的要求,项目需求分别为订单导入导出、与 OA 系统交互获取人员信息、订单流程预览、销售报表查询等,开发周期为 2 周.表 1 显示分别以 RAD 模型、增量模型、敏捷开发方法以及 ISI-RAD 模型为基础成立 4 个开发小组,每个小组分配 3 人,完成该业绩稽核功能的基本情况.

在表 1 的对比案例中,最先发布的是增量模型小组和 ISI-RAD 模型小组,但增量模型小组完成的并不是一个完整的模块,而是一个系统的初级样本.敏捷开发方法小组和 ISI-RAD 模型小组可以持续发布,每天系统都会有变化.综合来看,ISI-RAD 模型的优势较为明显,既包含增量模型的优先发布,又包含敏捷开发方法的持续集成.

RAD 模型的分领域开发能够确保开发人员之间的相互不干扰.相对而言,敏捷开发方法如果开发人员过多,详细设计将会非常零散,必须集结所有的人在一起才能完成,一旦有一个未确定,后期可能会引起蝴蝶效应,导致一系列 story 的修改.ISI-RAD 模型由于其结合了 RAD 模型的优势,从而规避了敏捷开发方法的这个缺点.

ISI-RAD 模型继承了敏捷开发的优势,对软件系统开发进度的掌控非常精细.相对于以往只能凭人力评估系统开发的大致进度而言,ISI-RAD 模型能够进行超细粒度的story掌控,并提供有力的数

表 1　对比案例

Tab.1　Comparative case

进度	RAD 模型	增量模型	敏捷开发方法	ISI-RAD 模型
第 1 天	按需求分成 4 个模块，每人选择 1 个，剩下的再分配，开始各自详细设计	选择“订单查询”、“订单流程查询”为第 1 期，集体讨论详细设计	将需求全部拆分为若干个 story，开始详细设计	按需求分成 4 个模块，选择“订单导入导出”为第 1 阶段，集体详细设计并拆分为多个 story
第 2 天	继续详细设计	分工编写代码	继续详细设计	编写各自 story
第 3－4 天	部分开始编写代码，部分还在讨论“OA 系统接口”	完成 1 期，确认“销售报表查询”为 2 期，继续	全部开始编写代码，搁置“OA 系统接口”	完成“订单导入导出”模块，确认“订单流程预览”模块，继续
第 5－7 天	开发中，没有完成 1 个模块	完成 2 期，确认“查询 OA 系统人员信息”、“订单流程预览”为 3 期，继续	开发中，没有完成 1 个模块.（story 都分在不同模块）	完成“订单流程预览”模块，确认“与 OA 系统交互获取人员信息”模块，继续
第 8－9 天	完成“订单导入导出”模块，开始“订单流程预览”	3 期中“OA 系统接口”阻塞进度	完成“订单导入导出”模块	还差 OA 系统接口，确认“销售报表查询”模块，继续
第 10－12 天	完成“销售报表查询”模块，协助其他成员开发	完成 3 期，剩下的确认为 4 期，继续	完成“订单流程预览”模块、“销售报表查询”模块	完成“与 OA 系统交互获取人员信息”模块，“销售报表查询”开发中
第 13 天	全部完成	继续 4 期	全部完成	全部完成（共 15 个 story）
第 14 天		全部完成		

据支撑和项目风险监控，监控报表如图 5 所示. ISI-RAD 模型中，所有 story 的状态都可以明确标识，方便工作任务的分配，因此在开发过程中即使人员有所变动，也只可能影响开发过程中的 1 个 story，而不会对开发的整个过程有太多影响.

同时，ISI-RAD 模型继承了 RAD 模型、增量模型、敏捷开发方法的优势，对比如图 6 所示（见下页）. 它摈弃了原有 RAD 模型需要足够人力资源和要求用户实现承诺的缺点，而保留了 RAD 模型在极短时间内创造出“全功能系统”的优势；它摈弃了原有增量模型容易退化为边做边改模型的缺点，而保留了增量模型可以快速交付的优势；它摈弃了原有敏捷开发方法无规划性的缺点，而保留了敏捷开发方法中项目进度及风险控制精细化的优势[19].

总story数	15				
KPI \ 状态	开发中	测试中	bug	二次bug	完成
数量	3	1	4	1	1
百分比	20%	6.67%	26.67%	6.67%	6.67%
比例	3	1	4	1	1
二次bug率	—	—	—	25%	—
在线率	—	—	—	—	53.33%
质量系数	—	—	—	—	0

项目监测表		作用
数量	当前状态存在的story个数	查看各状态数量
百分比	数量/总story数*100%	查看完成率
比例	各个状态的数量比	查看各状态负荷及空置率，把握平衡度
二次bug率	二次bug数/bug数*100%	查看bug修复的质量，单个story多次bug
在线率	(开发中+测试中+bug-二次bug+完成)/总story数*100%	查看story活动率，检测项目是否稳定
质量系数	1-bug数/(bug数-二次bug数+完成)	查看story完成的质量

图 5　ISI-RAD 模型中的项目风险监控报表

Fig.5　Project risk monitor sheet of ISI-RAD model

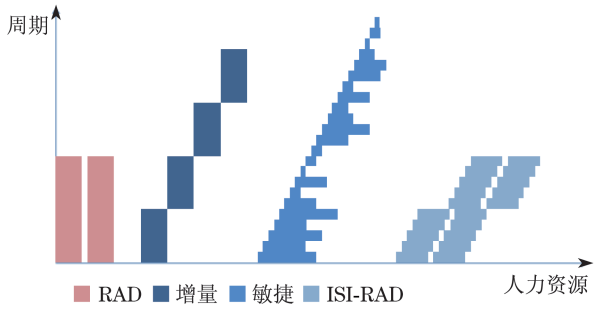

图6 ISI-RAD模型开发周期粒度对照表

Fig.6 Check list about development cycle size of ISI-RAD model

5 结束语

敏捷开发以快速反映及强调人员沟通为主要思想,但如果没有好的管理策略,类似的开发方法不可能起到好的作用.基于快速应用开发的ISI-RAD模型的提出,是解决这一问题的一种新思路,并经过实践检验,表现出十分强壮的生存力.

参考文献:

[1] Roger S P. Software engineering: a practitioner's approach[M]. New York: McGraw-Hill, 2010.

[2] Jorge A, Steve E. Anchoring and adjustment in software estimation[C]// Proceedings of the European Software Engineering Conference on the Foundations of Software Engineering. Lisbon: ACM Press, 2005: 5-9.

[3] Václav T R. Software engineering: the current practice[M]. Boca Raton: CRC Press, 2012.

[4] 盛钢,曹渠江.软件开发统一过程应用研究[J].上海理工大学学报,2003,25(1):94-98.

[5] Roberto Z, Jorge L N A. Project management model for a physically distributed software development environment[C]// Proceedings of the 36th Hawaii International Conference on System Sciences. Washington: IEEE Computer Society, 2003: 294-301.

[6] Pierre B, Robert D, Alain A, et al. Fundamental principles of software engineering——a journey[J]. The Journal of Systems and Software, 2002, 62(1): 59-70.

[7] Tsun C, Dac-Buu C. A survey study of critical success factors in agile software projects[J]. The Journal of Systems and Software, 2008, 81(6): 961-971.

[8] Tore D, Torgeir D. Empirical studies of agile software development: a systematic review[J]. Information and Software Technology, 2008, 50(9/10): 833-859.

[9] Jean-Guy S, Rajesh V. Agile practices in software development-experiences from student projects[C]// Proceedings of the 2006 Australian Software Engineering Conference. Sydney: IEEE Computer Society, 2006: 401-410.

[10] Frauke P. Requirements engineering in agile software development[D]. Germany: Fachhochschule Mannheim Diploma Thesis, 2003.

[11] Sudarsan R, Steven J F, Ram D S, et al. A product information modeling framework for product lifecycle management[J]. Computer-Aided Design, 2005, 37(13): 1399-1411.

[12] Toni S, Tomislav D. Rapid application development using software factories[DB/OL]. [2013-09-30]. http://arxiv.org/ftp/arxiv/papers/1201/1201.0853.pdf.

[13] Ken S. Agile project management with scrum[M]. Redmond: Microsoft Press, 2004.

[14] Alistair C. Using both incremental and iterative development[DB/OL]. [2013-09-30]. http://alistair.cockburn.us/Using+both+incremental+and++iterative+development.

[15] Outi S, Pekka A. Agile methods in european embedded software development organisations: a survey on the actual use and usefulness of extreme programming and scrum[J]. IET Software, 2008, 2(1): 58-64.

[16] 施佳,夏骄雄,张武.FSL-SP的研究[J].计算机工程,2007,33(8):182-184.

[17] Richard M S, Robert M N. Seven plus or minus two: a commentary on capacity limitations[J]. Psychological Review, 1994, 101(2): 357-361.

[18] Cleidson R B S, David R, Li-Te C, et al. Sometimes you need to see through walls—a field study of application programming interfaces[C]// Proceedings of the 2004 ACM Conference on Computer Supported Cooperative Work. New York: ACM Press, 2004: 63-71.

[19] Zheng L, Shardrom J, Gang W, et al. A hybrid model based on agile development[C]// Proceedings of the 2013 3rd International Conference on Electric and Electronics. Hong Kong: Atlantis Press, 2013: 149-153.

[20] Alessandro R, Andrea S. Concurrent object-oriented programming with agents[C]// Proceedings of the 2013 Companion Publication for Conference on Systems, Programming, Languages and Applications: Software for Humanity. New York: ACM Press, 2013: 89-90.

(编辑:董 伟)