

软件过程的比较框架研究^{*})

李松岭 金蓓弘

(中国科学院软件研究所软件工程技术中心 北京100080)

摘要 软件过程在软件开发中占有非常重要的地位。根据项目的需求和特点,选择一个合适的软件过程可以使项目的开发事半功倍。为了指导软件过程的选择,本文给出了一个简明的软件过程比较框架,并依据这个比较框架,对几个典型的软件过程进行了分析和比较。

关键词 软件过程, RUP, TSP, XP, 净室过程

Research on Comparison Framework for Software Process

LI Song-Ling JIN Bei-Hong

(Technology Center of Software Engineering, Institute of Software, Chinese Academy of Sciences, Beijing 100080)

Abstract Software process plays an important role in the development of software. A proper software process that chose according to the requirements and character of project will help you a lot. To guide the chosen of software process, this paper brings forward a frame, then states several software processes, at last analyzes and compares the software processes in the frame.

Keywords Software process, RUP, TSP, XP, Cleanroom

按 ISO/IEC 12207 标准,软件过程是指软件生命期中的若干活动(activity)的集合^[1]。活动,有时又称为工作流程(Workflow),又可细分为子活动或者任务。使用软件过程目的是为了在一定的时间和经费预算内开发出高质量的产品。一个好的软件过程可以提高软件开发组织的生产效率、提高软件质量、降低成本并减少风险。

近年来,对软件过程的研究呈现出两个特点:

(1)研究层次的变化,从高层抽象全面过渡到对过程细节的研究。早期,主要针对抽象的软件生命期模型进行软件过程研究,过程活动的细节主要取决于个人的素质和经验。目前对软件过程的研究已全面过渡到过程细节的研究。

(2)研究内容的扩展,研究范围已发展到对“过程建模”、“过程实施”、“过程度量”、“过程改进”等诸多相关内容的研究。

早期的软件过程有瀑布式、增量式、螺旋式(此处指应用瀑布模型、增量模型、螺旋模型这些过程模型的具体过程)等,最近几年还在不断地推出新的软件过程,如 RUP(Rational Unified Process)、XP(eXtreme Programming)、TSP(Team Software Process)、净室(Cleanroom)过程等。

对软件过程进行全面的分析、比较,可为过程选择提供指导,更进一步,这种比较将有助于过程工程师进行过程改进,甚至设计新的软件过程。为此,本文就软件过程的比较框架进行了研究。

1 软件过程比较框架

我们抽取软件过程的以下要素用于软件过程的比较:软件过程所包含的活动、活动执行的方式、要交付的工件(artifact)、参加的角色、支持工具、与 CMM(Capability

Maturity Model)的关系、适用的范围。

1.1 软件过程所包含的活动

软件过程最起码要包含支持软件整个生命期的活动。划分软件生命周期的方法有许多种,软件规模、软件类型、软件开发时使用的方法及开发环境等,都影响软件生命周期的划分。基本的生存周期包括软件计划、需求分析、设计、编码、测试和维护。相应地,有计划、需求分析、设计、编码、测试和维护活动。在不同的过程中,活动的抽象程度和可操作程度也不同:有的过程给出了具体的操作步骤,有的过程仅给出一些活动要点。

此外,软件过程所包含的活动还受下列因素的影响:

(1)所采用的软件开发方法。例如,采用组件化开发方法,会包含下列活动:组件资格认定,用于发现和分析哪些组件能满足应用需求;组件改编,用于改写组件以满足体系结构需求;组件组合,用于给出组件连接和协调机制;组件更新,指组件的版本升级,以便适应系统需要。

(2)项目对软件质量的要求。对高质量软件的追求是过程发展的原动力,从宏观看,过程的标准化是质量保证的手段之一;从微观看,过程还可拥有具体的手段来改善和保证软件质量。例如,在软件开发中,经常进行评审活动,用于评价软件开发的状态和质量。

(3)项目管理要求。通过项目管理,可以平衡竞争目标、管理风险并克服制约因素,从而最终成功交付同时满足客户和用户双方需要的产品。软件过程不仅要解决技术问题,还要解决管理问题,现在项目管理也常常是过程的一部分。

1.2 活动执行的方式

活动的执行方式包括活动何时执行、有哪些行为约束条件、怎样提供反馈等。执行方式可以是线性顺序的、迭代的、并

^{*}本文研究得到国家自然科学基金(60103008)的资助。李松岭 硕士研究生,研究方向:分布式计算,软件体系结构和过程;金蓓弘 博士,副研究员,研究方向:软件工程技术,分布式计算。

行的、嵌套的或是有条件触发的。比较典型的有线性顺序的执行方式和迭代的执行方式。

1.3 要交付的工件

工件是项目期间生成并使用的最终或中间产物。一个工件是描述活动所产生信息的实体,包括文档、模型、代码等。最终交给用户的工件应该是系统以及系统的说明性文档。

毫无疑问,所有的软件过程都产生工件,用于获取和传达项目信息。产生什么样的工件,什么时候产生,由谁产生,哪些是必须的,哪些是可选的,不同的软件过程对于这些问题的回答都有差别。

1.4 参加的角色

角色是过程的执行主体,由角色完成软件过程的活动,制造各种工件。角色可能是一个人,也可能是一组人。一个人或一组人也可能扮演多个角色。

软件过程一般会包含的角色有项目管理人员、系统分析员、开发人员、最终用户。不同的过程会根据自身的特点和强调的重点,增加特定的角色,例如强调复审的重要性,可增加多种复审人员如需求复审者,体系结构复审者等等。

1.5 工具支持

工欲善其事,必先利其器,一个好的软件过程必定有其支持工具。角色执行活动,活动产生工件,工具则可以使角色更好地执行活动。有工具支持的过程才能称作是一个完善的过程。

1.6 与 CMM 的关系

CMM 即过程能力成熟度模型,是美国卡内基梅隆大学软件工程研究所(SEI)在美国国防部的资助下于20世纪80年代末提出的,用于评价软件机构的软件过程能力成熟度。根据过程成熟度的不同,CMM 划分成由低到高的五个级别:初始级、可重复级、已定义级、已管理级和优化级。考察软件过程覆盖了哪些 KPA(Key Process Area),能达到 CMM 的哪一个级别,对项目执行组织选择过程、改进过程有直接的指导意义。

1.7 适用范围

对于一个软件过程而言,适用范围包括项目的规模、项目的性质。描述软件规模的度量参数有源代码行(LOC)、参加人员数和研制期限。按照通常的衡量软件规模大小的尺度,5千行以内是小程序设计,1~5万行代码的软件是小规模软件,5~10万行的软件是中规模软件,10~100万行的软件是大规模软件,100万行以上是超大规模软件^[2]。项目的性质主要是指对项目的熟悉程度。项目是否为人熟知,是否已经存在类似系统,这些对项目的开发都会产生影响。项目的性质还包括需求是否明确。需求明确的系统相对来说容易开发,需求不明确的项目就需要软件过程中有一定的策略来保证项目的开发。任何一个软件过程都不是万能的。一个软件过程只能在一定的范围内是适合的、有效的。换了一个环境,在另一个范围内,同一个软件过程可能不但效率低下,还有可能产生出劣质的产品。

2 典型软件过程

瀑布式与螺旋式是早期提出的软件过程,虽然存在缺陷,但是直到现在仍然有很多应用系统的开发采用它们作为软件开发的过程;RUP 是由提出 UML 的三位方法学家 Booch、Rumbaugh 和 Jacobson 提出的,所以自从问世以来备受关注;XP 是以敏捷著称的方法,通常被认为是敏捷过程(Agile

Process)的代表;TSP 是以工作组形式进行开发,特别适合中国目前软件开发的现状;而净室过程基于数学分析和统计,是一种形式化的软件过程。本文选择了这几个典型的软件过程进行分析和比较。

2.1 瀑布式

Winston Royce 在1970年提出了著名的“瀑布模型”,包括如下六个工程活动:制定计划、需求分析和定义、软件设计、程序编写、软件测试、运行和维护,并规定了它自上而下、相互衔接的固定次序,如同瀑布流水,逐级下落^[3]。遵循瀑布模型的开发活动一般具有以下特点:(1)从上一项开发活动接受该项活动的工作对象,作为输入。(2)利用这一输入,实施该活动应完成的开发活动。(3)给出该项活动的工作成果,作为输出传给下一项开发活动。(4)对该项活动的实施工作成果进行评审。若其工作成果得到确认,则继续进行下一项开发活动,否则重作该项活动,甚至返回前项的活动。尽量减少多个阶段间的反复,以相对来说较少的费用来开发软件。

2.2 螺旋式

Barry Boehm 在1988年正式发表了软件系统开发的“螺旋模型”。螺旋模型的特点是增量和迭代。螺旋开发过程由以下几个步骤组成:(1)计划:确定目标、方案和限制。(2)风险分析。(3)实施工程:进行开发,产生一个原型系统。(4)就当前目标对原型系统进行验证。(5)增加新的功能,重复以上操作。每经历了瀑布模型的各个步骤就会产生一个原型,每次新的版本都是对旧的版本的升级。

螺旋模型是将瀑布模型和演化模型等结合起来,强调了在此之前其它模型忽略了风险分析^[4]。需要具有相当丰富的风险评估经验和专门知识,才能较好地发挥这种开发过程的特点。

2.3 XP

20世纪90年代初,Kent Beck 和 Ward Cunningham 开始试验一种新的软件开发思路——怎样使每件事情都更简单更有效,最终形成了一种新的软件开发过程 XP。XP 是一个轻量级的软件开发过程。XP 强调沟通(Communication)、简单化(Simplicity)、反馈(Feedback)和勇气(Courage)^[5]。XP 认为,人员之间的交流、代码交流是最重要的,而文档是第二位的,所以要尽量少产生文档;XP 认为,复杂的事情不容易做好,而做简单的事情则容易得多。XP 的做法是先做好一件简单的事情,以后在需要时进行小的修改。反馈是和交流与简单性一起发挥作用。XP 中的勇气不仅包括工作的勇气而且包括抛弃的勇气。XP 的这四个特点是相互作用的:交流充分,任务就明确,系统会更简单,因此需要的交流也会变少;反馈越多越容易进行交流;充分的交流和系统的简单性有助于提高开发人员的勇气,有了勇气,会增加把事情做得简单的信心,而反馈又巩固了勇气。

2.4 TSP

Watts Humphrey 1989年制定了 PSP(Personal Software Process),PSP 是一个良定义的个人软件开发过程。在此基础上,Watts Humphrey 制定了 TSP。TSP 是一个良定义的、可度量的过程,一个软件项目组可以用 TSP 计划工作、实现计划并且不断改善软件开发过程。TSP 为软件的开发提供了有规律的严格的框架。一个小组至少要有两个成员,分别承担一定的角色。小组的成员为了一个共同的目标努力,并在工作的过程中互相支持、互相帮助。开发 TSP 的主要动机是为了使小组成员的能力和力量能联合起来组成一个整体,使之超

过个体力量之和,从而能更有效地产生出高质量的软件产品。实施 TSP 的前提条件是小组成员是有相应技术的并经过一定的训练^[6]。

2.5 RUP

RUP 是由 Booch、Rumbaugh 和 Jacobson 以 Rational 公司的过程模型 Objectory 为核心提出的软件过程模型。RUP 是一个重量级的软件工程过程,它提供了严格的方法在开发组织内分配任务和责任,它的目标是按照制定的时间计划和经费预算,开发出满足最终用户需求的高质量的产品。

RUP 最重要的目标是降低项目的风险。RUP 有四大显著特点:

- 用例驱动:在 RUP 中,用例在多个环节发挥了作用。用例的概念可用来表示业务流程,在识别需求时,用例描述所要求的功能,并由客户确定这些功能;然后分析和设计用例,并实现用例;在测试阶段,由用例构造测试用例,对系统进行验证;最后,在项目管理工作流程中,用例被用来作为计划迭代式开发的基础。

- 以体系结构为中心:在整个生命周期中,设计活动是以体系结构为中心展开的。流程初期迭代的重点在于生成并验证软件的体系结构,最初的开发周期要形成一个可执行的体系结构原型,然后在以后的迭代中逐渐演变成最终系统。

- 迭代和增量开发:RUP 在迭代中,采用迭代和增量开发。每个迭代都会有一些新的特征加入,使开发出的产品不断成熟。迭代和增量开发更能适应变更,开发风险更小。

- 动态可适应性: RUP 在提供非常丰富的过程内容的同时,也提供了可裁减性。RUP 可以去掉项目不必要的工件和工作流,吸收若干 XP 如使用用例卡片进行交流、让客户参与开发、短小的迭代周期、结对编程、程序重构等。

2.6 净室过程

净室方法的理论基础建立于20世纪70年代末80年代初,资深数学家和 IBM 客座科学家 Harlan Mills 阐述了将数学、统计学及工程学上的基本概念应用到软件的设想。净室的哲学观念是建立一种能排除产品缺陷引入的使得开发时间和开发代价趋向合理和有效的构造方法^[7]。按净室方法,不是先制作一个产品然后去消除缺陷,而是要求在规约和设计中消除错误,然后以“干净”的方式开发软件。

净室过程强调在规约和设计上的严格性;强调使用基于数学的正确性证明来对设计模型的每个元素进行形式化验证;强调统计质量控制技术,包括基于客户对软件的预期的使用的测试;强调缺陷的预防而不是如何去除缺陷。净室过程的一个主要目标是产生零缺陷的软件。净室过程可用如下三个关键技术来刻画:置于统计过程控制之下的增量开发,基于函数的规约设计和验证以及统计测试和软件认证。

3 软件过程比较

下面我们根据软件过程比较框架的各要素,对上述典型的软件过程进行比较。

3.1 软件过程所包含的活动

瀑布式和螺旋式只支持了软件开发基本的活动,而且只给出了一个抽象框架,没有进一步的分解和细节的描述。各项活动是编程人员共同来完成。这两种过程由于抽象程度过高,因此可操作性比较差。

XP 只描述了四种基本活动:编码、测试、倾听、设计。XP 活动由一系列的任务来完成:小的发布、隐喻、简单的设计、代

码重构、结对编程、代码集体拥有、经常集成等。XP 对这些任务的描述都很简单,例如要求在编码中进行结对编程,但没有说明如何在实际环境中两人一组进行工作的细节。XP 没有专门的设计阶段,设计是融合在编码当中的。XP 用代码重构来保证系统的基础构造。XP 十分注重测试,这也是程序健壮性的保证条件。

TSP 的活动包括启动、策略、计划、需求、设计、实现、测试和后期维护。TSP 强调计划的重要性,每个阶段都有详细的计划。TSP 有一个启动阶段,这个阶段起承上启下的作用,以达到持续开发的目的。TSP 的活动和角色联系密切,每个活动根据角色进一步划分成任务,每个角色分担一定的任务。TSP 的后期维护中有一个极具特色的子活动是评估。TSP 的评估不仅要对项目进行评估,还要对小组以及小组中的每个成员进行评估。

活动在 RUP 中被称为工作流程,RUP 共有9种工作流程:业务建模、需求、分析与设计、实现、测试、部署、配置与变更管理、项目管理、环境,其中,有几个工作流程是有别于其它过程的:

- 业务建模:该工作流程用于了解目标组织的结构及机制,了解目标组织中当前存在的问题并确定改进的可能性,确保客户、最终用户和开发人员就目标组织达成共识,是需求工作流程的一种重要输入。

- 项目管理:该工作流程为管理风险提供框架,并监测迭代的进度。

- 配置与变更管理:在项目的开发人员众多,并生成大量工件时,实施该工作流程有助于避免混乱情况的发生。

XP 采用面向对象开发方法,XP 对管理的要求不太严格,活动是在简单性前提下展开的。与之相比,RUP 对任务的划分、活动的展开提供了很好的指南,对质量和风险都作了很好的控制。

净室过程一共有14项活动,根据管理、规约、开发和认证分类如下:

- 管理过程包括项目计划、项目管理、性能改进、工程变更。

- 规约过程包括需求分析、功能规约、使用规约、结构规约和增量计划。

- 开发过程包括软件再工程、增量设计、正确性验证。

- 认证过程包括使用建模、测试计划、统计测试和认证。

净室过程中有些特有的活动,例如,结构规约活动跨越整个生命周期,定义了体系结构和策略;功能规约和使用规约活动对需求定义做精确描述;软件再工程活动在一个增量中为使用准备需要的软件。增量设计和正确性验证用来为一个增量开展设计和编码,并认证其正确性;统计测试和认证用于评价增量是否好用。

3.2 活动的执行方式

瀑布式过程中,各个工程活动按典型的线形顺序执行。这种执行方式不支持重用,后期的变化、迭代、改动相对困难,而且,没有一个联系各个阶段的统一模型。

螺旋式、XP、TXP 和 RUP 按迭代方式执行。一个理想的 XP 生命周期为探测-计划-第一个发布的迭代-产品化-维护-死亡。产品化阶段是由多个迭代组成的,一般一个迭代为2~6周。TSP 的开发周期一般分为几个迭代,每个迭代有四个阶段:阶段1包括启动和策略,阶段2包括计划和需求,阶段3包括设计和实现,阶段4包括测试和后期维护。RUP 对迭代的生命

周期进行了改进,根据时间和 RUP 的核心工作流划分为二维空间。按照时间划分为四个阶段:先启阶段、精化阶段、构造阶段、产品化阶段。每次迭代要经过四个阶段,每个阶段都可选择进行九个核心工作流程。每个阶段都有一个里程碑,满足里程碑要求,就结束本阶段。其中产品化阶段的里程碑还标志着一次迭代的结束。

净室过程采用增量的开发,并置于统计过程控制之下。一般经过需求收集、盒结构规约、形式化设计、正确性验证、代码检查、统计性使用测试、认证。当通过认证后才可以引入下一个增量,再经过如上步骤进行开发。

3.3 过程中的角色

瀑布式和螺旋式过程中,角色并没有明确的分工。所有的活动都是由开发人员来完成。

XP 中的角色有编程人员、客户、测试人员、跟踪人员、教练、顾问、老板。XP 中有几个角色与别的过程不同,这也是 XP 的特色。其中,客户总要在现场和开发人员一起工作,负责写用户素材(User Story),完成项目的需求,这种工作方式大大简化了需求分析。编程人员采用结对编程,以减少编码缺陷。教练负责指导项目组的工作,当项目组偏离轨道时提醒他们,当所有人都惊慌时,教练必须保持冷静。顾问提供项目的技术支持。教练和顾问是 XP 简单性前提下能够保持正确性的重要角色。

TSP 中的角色有小组领导、客户接口经理、设计经理、实现经理、测试经理、计划经理、过程经理、质量经理、技术支持经理,其他角色可根据需要添加^[4]。TSP 的角色根据开发时的活动而设置,例如计划有计划经理,设计有设计经理等。每个经理负责自己经理角色的同时还要承担开发人员角色,进行系统的开发。

RUP 定义了30种角色,这使得 RUP 的分工非常明确。RUP 强调复审的重要性,增加多个复审人员:业务模型复审者、需求复审者,体系结构复审者等。RUP 强调对变更的控制,故增加了变更控制经理角色。RUP 的一个重要角色是体系结构设计师,他负责在整个项目中对技术活动和工件进行领导和协调,有项目经验、有领导才能和善于沟通的人才能胜任此角色。与其他角色相比,体系结构设计师的见解重在广度,而不是深度。

净室过程是面向工作组的,由小组进行开发。一个项目可能会有多个小组,每个小组一个组长。由组长负责分配任务。组之间必须有协调人员进行工作的协调。有两个特殊角色:规约人员和认证人员,规约人员用盒规约方法来描述功能规约,认证人员用数学的方法进行测试并计算认证可靠性。

3.4 过程产生的工件

瀑布式和螺旋式过程会产生需求文档、分析文档和设计文档和测试文档等。

XP 产生的工件较少,只包括一些必须的文档和代码,因为它强调用代码进行交流。XP 中的工件也与其他软件过程中的工件有很大区别,例如一般过程中都有需求文档,在 XP 中是用客户书写的用户素材来代替的,需求被写成小的简单的故事。

TSP 产生的工件包括文档和大量的表格。除了常规的需求文档、设计文档、测试文档、检查文档,TSP 有一个缺陷日志,用来记录过程中的缺陷。TSP 的特色工件是各种表格,包括从项目开始的 Start 表、每周例会的 Week 表、时间问题和错误的日志,到计划、质量、任务等的总结,通过这些表格,角

色可以顺利地开展工作,完成每个阶段的任务。

RUP 充分考虑了大规模甚至超大规模项目开发所需要的细节,它描述了各个阶段产生的100多种工件,不仅包括计划和文档,还包括用例模型、设计模型等。但在按 RUP 开发具体项目时,并不一定要产生所有的工件,过程工程师可以有针对地选择和裁减工件。

净室过程重要的工件是模型,好的模型是开发的良好基础。例如认证有三个模型:取样模型、构件模型和认证模型。取样模型中软件测试执行 m (由数学方式导出的值) 个随机测试用例,构建模型对一个由 n 个构件组成的系统进行认证,认证模型中系统的整体可靠性被规划和认证。

所有的软件过程都产生的一个工件就是代码,每个过程都强调代码的正确性。代码在过程中的作用也不相同,例如,XP 中,代码在开发过程中起到了交流的作用。

3.5 工具支持

随着软件过程的发展,过程的支持工具经历了一个从无到有、从抽象到具体的过程。参见图1。

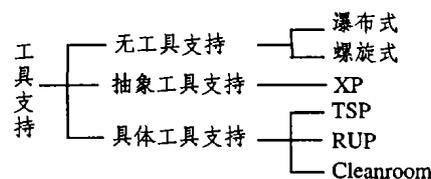


图1 软件过程的支持工具

瀑布开发方法与螺旋开发方法没有工具的支持,主要靠编程人员的经验进行开发。

XP 的支持工具是极端(extreme)的理念:尽量把每件事都做得简单,需要做计划时可能所有的人去做计划,测试时可能所有的人都在测试。顾问保证了这个理念,当出现技术难题时,顾问就会发挥作用、解决问题。

TSP 的支持工具是一系列的草案和表格。草案包括设计草案、开发草案、实现草案、检查草案、启动草案、计划草案、后期维护草案、需求草案、软件配置管理草案、策略草案、集成和系统测试草案、单元测试和测试方案草案、每周例程。每个草案都描述了该阶段应该做什么、由谁去做、产生什么工件。例如需求草案,列出了开始条件、任务的更新(开发经理)与分配(小组领导)、系统测试计划(开发经理)、更新检查(质量经理)和结束标准,产生需求文档、检查草案、工程记录手册等工件。

支持 RUP 的工具种类详细而且功能强大,例如,需求管理工具有 Rational RequisitePro;可视化建模有 Rational Rose;编程有 Rational Apex/Ada、Rational Apex/C++;配置管理有 Rational ClearQuest;文档生成工具有 Rational SoDA/Word 和 Rational/Frame;自动测试工具有 Rational Robot、Rational TestFactory 等。这些支持工具大大方便了项目的开发。

净室过程以数学分析和统计作为基础,例如测试工具 CleanTest 给出了一个友好的图形用户界面,是一种基于统计的测试。所使用的测试用例强调软件的“黑盒”行为,并由统计的方法确定。

3.6 与 CMM 关系

图2给出了过程与 CMM 的关系。瀑布式、螺旋式和 XP 覆盖了2级和3级的大部分 KPA,RUP 覆盖了2级和3级的 KPA,TSP 几乎覆盖了所有的 KPA^[9],净室过程覆盖了所有的 KPA。

成熟级别	关键过程区域	瀑布式	螺旋式	XP	TSP	RUP	净室过程
2级 可重复级	需求管理	✓	✓	✓	✓	✓	✓
	项目策划	✓	✓	✓	✓	✓	✓
	项目的追踪	✓	✓	✓	✓	✓	✓
	子合同管理	×	✓	×	×	✓	✓
	软件质量保证	✓	✓	✓	✓	✓	✓
	软件配置管理	✓	✓	✓	✓	✓	✓
3级 已定义级	机构过程焦点	✓	✓	✓	✓	✓	✓
	机构过程定义	✓	✓	✓	✓	✓	✓
	培训管理	×	×	×	×	✓	✓
	集成软件管理	✓	✓	×	✓	✓	✓
	软件产品工程	✓	✓	✓	✓	✓	✓
	组间协调	✓	✓	✓	✓	✓	✓
4级 已管理级	同行评审	✓	✓	✓	✓	✓	✓
	过程的量化管理	×	×	×	✓	×	✓
5级 优化级	软件质量管理	×	×	×	✓	×	✓
	过程变更管理	×	×	×	✓	×	✓
5级 优化级	技术变更管理	×	×	×	✓	×	✓
	缺陷预防	×	×	✓	✓	×	✓

✓: 表示覆盖 ×: 表示未覆盖

图2 典型过程与CMM的关系

3.7 适用范围

在瀑布开发方法中,只有到生命周期的后期,才能确知周围是否存在风险。瀑布开发不适合需求模糊的系统,显然也不适宜用来开发大型项目,因此瀑布开发主要应用在对于需求和需求的实现有很好的理解的情况。只有已经存在类似系统的情况下,采用瀑布开发才有一些优势。

螺旋开发方法虽然已经对瀑布开发方法进行了改进,但是由于没有明确的角色分工,仍然难以处理大的复杂的项目。

XP灵活多变,适合需求不清的项目。但是由于受开发活动的限制例如代码集体拥有、同一个房间里工作以及开发人员数量不能过多,因此XP不适用于大型的项目。

TSP的小组可以有2至20人,一般在10至12人,这样的小组效率最高,最能够发挥个人与小组的能力,适应能力也比较

强。TSP适合开发小型、中型的项目。也可以组成一个多小组的TSP过程,例如包含150人,这样的组可以处理大规模的项目。

RUP定义了大量的角色、工件和活动,适用于大规模和超大规模项目的开发,由于RUP有上述诸多优点并且提供了需求管理和变更管理的工具,故RUP能够适应开发中的各种变化,并且使开发的风险总在控制之下。另一方面,RUP又是可裁剪的,可按项目需要减少角色、工件和活动,剪裁之后的RUP是一个类似XP的过程,虽没有XP那样简单,但是风险性较小。

净室过程通过数学分析使开发中二义性、不完整性和不一致性更容易被发现和纠正。但是开发费用高,需要背景知识和培训,与用户通信也比较困难。净室过程适用于开发要求严格的关键软件,以及如果发生错误会造成严重经济损失的场合。

小结 本文提出了一个简明的软件过程比较框架,框架主要包括六个方面的内容:软件过程中的活动、活动的执行方式、过程中的角色、过程产生的工件、工具支持、与CMM的关系和适用的范围。在此框架基础之上,我们对几个典型的软件过程进行了分析和比较。这些比较将有助于我们选择、改进已有过程,同时,也为新型过程的设计提供了一个基础框架。

参考文献

- 1 朱三元,钱乐秋,宿为民. 软件工程技术概论. 科学出版社,2002
- 2 冯玉琳,黄涛,金蓓弘. 网络分布计算和软件工程. 科学出版社,2003
- 3 周之英. 现代软件工程. 科学出版社,1999
- 4 郑人杰,殷人昆,陶永雷. 实用软件工程. 清华大学出版社,1997
- 5 Beck K. Extreme Programming Explained. Addison-Wesley, 2000
- 6 Humphrey W S. 小组软件过程. 人民邮电出版社,2000
- 7 Pressman R S. 软件工程实践者的研究方法(第4版). 机械工业出版社,1999
- 8 Humphrey W S. The Team Software ProcessSM (TSPSM). (CMU/SEI-2000-023)
- 9 McAndrews D R. The Team Software ProcessSM (TSPSM): An Overview and Preliminary Results of Using Disciplined Practices. (CMU/SEI-2000-TR-015)

(上接第71页)

对中间表示树进行中根遍历,根据3.2.1节中的映射定义从节点上或数据集逐行取得数据输出到XML文档,即可实现从关系数据到XML文档的转换,生成的XML文档结构与中间表示树的结构一致,生成的XML文档即XQuery查询的最终结果。

4 查询的优化

对于XML数据树的查询,可以在搜索空间、代价估算、计划枚举等三个方面进行优化,但是在VegasXML中,数据的查询最终是由SQL实现,查询的对象并不是XML文档,而是关系数据库,因此查询的效率仍然取决于DBMS,主要的优化工作也是由DBMS完成的。

结语 VegasXML为用户提供了虚拟的XML数据库,并且实现了XQuery的查询功能,利用了关系数据库成熟的管理技术,为面向Web的数据库应用提供了新的选择。但是由于关系模型本身的缺陷,一些半结构化数据无法映射,因此该系统适合于数据结构不太复杂的应用,仍然有很多问题需要研究。今后将在这些方面对系统作出改进,并期待W3C在XQuery中增加对数据库更新的规范,从而实现一个完整的

XML数据库管理系统。

参考文献

- 1 Staken K. Introduction to Native XML Databases. <http://www.xml.com/>
- 2 Goldman R,McHugh J,Widom J. Lore: A Database Management System for XML. Dr. Dobb's Journal April 2000
- 3 XML Path Language (XPath) Version 1.0 W3C Recommendation. 16 Nov. 1999. <http://www.w3.org/TR/xpath>
- 4 XQuery 1.0: An XML Query Language W3C Working Draft. 15 Nov. 2002. <http://www.w3.org/TR/xquery/>
- 5 Ceri B. Comparative Analysis of Five XML Query Language-s. SIGMOD Record, 2000, 29(1): 68~79
- 6 Abiteboul S,Quass D,McHugh J, Widom J,Wiener J. The Lorel Query Language for Semistructured Data. International Journal on Digital Libraries, 1(1): 68~88
- 7 硕网资讯. 洞悉XML. 北京大学出版社,2001
- 8 王照岳,孙建伶,董金祥. XML数据库管理系统研究. 计算机科学, 2002, 29(1): 115~117
- 9 郑仕辉,周傲英,季文赞,等. 基于SQL的XML查询的有效实现. 计算机研究与发展, 2001, 38(4): 422~428
- 10 Zhang Xin, et al. I Shrank the XQuery! — An XML Algebra Optimization. Approach Workshop on Web Information and Data Management (WIDM'02), Nov. 2002
- 11 Zhang Xin. Rainbow: Relational Database Auto-Tuning for Efficient XML Query Processing. A Proposal for a Dissertation in Computer Science. May 2001