浅析统一软件开发过程

季建华[1],宋 禾[2],杨 艳

(1. 西昌学院,四川 西昌 615013;2. 西昌市畜牧局,四川 西昌 615000)

【摘 要】本文对统一软件开发过程的发展历程,统一软件开发过程的用例驱动、以架构为中心、迭代和增量进行阐述,对统一软件开发过程的优缺点进行了讨论。

【关键词】用例驱动;架构;迭代;增量

【中图分类号】TP311.52 【文献标识码】A 【文章编号】1673-1891(2006)03-0061-03

1 统一软件开发过程的发展历程

20世纪60年代软件危机的出现,使人们认识到,大型的、复杂的软件系统的开发是一项工程,必须按工程学的方法组织软件的生产与管理,必须经过分析、设计、实现、测试、维护等一系列的软件生命周期阶段,而这一过程的有效性决定了软件产品的质量,或者说过程的质量决定了产品的质量。

同时人们也认识到,在软件开发过程中,编程固然重要,但更具有决定意义的是系统分析、设计并建立系统模型。在一个软件系统构建以前,先为它建立模型,其意义就像在建筑楼房前先要画图纸(blueprint)。随着软件系统复杂度的日益增加,好的建模技术越发显得重要,当然,好的建模语言及标准则是建模技术的关键要素。

20世纪80年代,随着面向对象方法在编程领域的兴起和走向成熟,面向对象方法开始向系统分析、设计阶段延伸。截至1994年,公开发表并具有一定影响的OOA&D方法已达50余种,同时,出现了相应的建模方法和语言。这众多的OOA&D及建模方法虽然采用的主要概念与原则大体上是一致的,但其差异却给用户的选择带来了困惑。所以,形成一种统一的面向对象软件开发过程、方法及建模方法和语言,势在必行。

90年代中期,随着Grady Booch、Jim Rumbaugh以及Ivar Jacobson陆续加入Rational公司,他们开始把各自的方法统一起来,1995年10月发布了第一个版本,称作"统一方法"(Unified Method 0.8),但鉴于这个统一过程的产物主要是一种建模语言,所以从1996年6月发布第2个版本开始改称"统一建模语言"(Unified Modeling Language),即UML 0.9。1997年1月,UML被提交给OMG(Object Management Group)申请标准化。1997年11月,OMG采纳UML 1.1作为基于OO技术的软件开发标准建模语言。现在,UML为OMG所有,2003年3月发布了一个标准版本是UML 1.5。

UML是一种用于对软件密集型系统及其制品(artifacts)进行可视化(visualizing)、详述(specifying)、构造(constructing)和文档化(documenting)的建模语言,尤其适用于分析、设计阶段的软件系统建模。它的语义、表示法和使用规则的定义完整而详细,表达能力丰富;作为一种工业标准的建模语言,它得到了众多软件公司、工程师的认可和支持;它还不断融入软件工程领域的新思想、新方法和新技术,并逐步完善。

统一软件开发过程经历了多年的发展和实际运用,并从众多的来源中获得灵感、取长补短,其定义和规范不断改进、完善,包括引入UML建模语言和方法;其目标是希望通过这样一个过程,开发出健壮的、有弹性的和可扩充的软件系统。目前,统一开发过程已经发展成熟,被广泛采用,最经典产品是Rational公司的Rational Unified Process。

2 统一软件开发过程

收稿日期:2006-09-25

作者简介:季建华(1966-),男,高级实验师,主要从事高校现代教育技术的实践与应用研究。

2.1 统一软件开发过程概述

统一过程是一个通用的过程框架，可用于各种软件系统、各种应用领域、各种组织机构、各种功能级别以及各种项目规模。

统一过程使用UML来制定软件系统的蓝图。

统一过程基于面向对象技术和基于component（构件、组件或部件）。

统一过程的突出特点为：用例驱动、以架构为中心、迭代和增量的。

2.2 统一过程是用例驱动的(use case driven)

软件系统是为用户服务的。为了构造一个软件系统，必须了解用户需要该软件为他做什么、提供什么功能、具有什么性能、用户会怎么使用该软件等等。也就是要获取用户需求。

为了描述用户需求、方便和用户沟通，于是我们描述用例(use case)。简单讲，一个用例就是用户与计算机之间的一次典型交互活动，或者说是一组动作序列，并且要有结果。为了让这种用例的描述更直观，于是我们画用例模型图(use case diagram)。通过用例图描述典型的使用情景来展示需求，方便理解。

2.3 统一过程是以架构为中心的（architecture - centered）

架构刻画了软件系统的整体设计，子系统或构件的集成与相互关联，突出了系统的重要特征。

架构的设计依赖于设计者的知识和经验，依赖于设计者对全局/细节、重点/非重点的判断。当然，架构的设计也受其他许多因素的影响，如：软件应用平台、项目实施条件、系统运行环境、是否要与遗留系统集成、是否有可重用构件或模块、非功能需求等。

当需求确定以后，就可以设计架构了。首先创建和用例不太相关的架构的轮廓；其次，考虑重要的用例(代表系统主要功能)，丰富架构设计；把更多的用例考虑进来，完善架构设计，直到架构稳定；同时，也在架构设计中完善用例模型。

2.4 统一过程是迭代（iterative）和增量(incremental)的过程

对于一个大型、复杂的软件项目，我们不太可能一次把它全部完成。这就需要从一个可行的局部开始，逐次增加（当然也可能是替代）和完善，这就是迭代和增量的开发过程。当然，这种迭代和增量的过程必须是按照计划好的步骤实施的，也就是受控的。

开发人员基于两个因素来确定一次迭代：首先，一次迭代过程要实现一组用例；其次，迭代过程要解决最突出的风险问题。

在一次迭代的末期，要验证目标是否实现（用例是否被实现、风险是否被降低）。如果目标达到，则开发工作进入下一次迭代；如果未能达到目标，则需要找出问题所在，考虑解决它。

由于不可预见的问题可能会对受控迭代过程产生较大的影响，譬如：加插迭代过程或改变迭代过程的顺序，所以，如果在对迭代过程进行计划的时候，能把不可预见的问题减到最少，则可增加受控迭代过程顺利实施的机率。

2.5 统一过程

从生命周期的角度看，统一过程实际上是一系列的循环，每个循环结束时都应该向用户提交一个产品版本。

每个循环包含四个阶段：初始(inception)、细化(elaboration)、构造(construction)、移交(transition)。每个阶段又可能包含一个或多个迭代过程。而一次迭代过程可能经历全部五种工作流（需求、分析、设计、实现和测试）或部分工作流。

2.5.1 初始(inception)

获取需求，估计项目成本、风险，初步的可行性判断，建立业务模型、简化的用例模型，勾画架构的轮廓；规划下一阶段的工作。

2.5.2 细化(elaboration)

详细说明绝大多数用例，设计系统的架构、确定架构的基线（baseline），规划完成项目的工作任务、进度，确定风险是受控的，确定用例、架构、计划是稳定可靠的。

2.5.3 构造(construction)

基于架构基线开发出较完善的系统(architecture - centered)，产品应满足用户需求，实现开了方和客户达成了共识的所有用例，产品经过了开发方的测试。

2.5.4 移交(transition)

制作产品并移交用户，提供用户培训和技术支持；用户试用并报告产品的缺陷和不足；维护部门会同开发人员会立即更正那些对最近一个版本的操作会产生重要影响的缺陷，另一些缺陷的更正或用户的改进建议则会在下一个正式版本中体现出来。

3 统一软件开发过程的优缺点

3.1 优点

迭代式开发方法是一个不断降低风险的过程，每一次迭代过程都选择风险最大的 UseCases 执行。因此风险在迭代过程中不断地被发现、被消灭。

迭代式开发方法能够更容易地管理需求的变化，整个开发过程由一次次的独立迭代组成，项目经理能够比较容易地调整迭代过程，使最终产品满足变化的需求。开发人员以及项目相关人员能够及时地从迭代过程中得到反馈信息，并能够及时修改以前工作中的失误，有效地监控开发过程，并对迭代工作流进行校正，这对一个时间跨度很长的项目具有重要的意义。

以 UseCase 驱动、体系结构为中心的开发使得开发人员能比较容易地控制整个系统的开发过程，管理其复杂性并维护其完整性。

体系结构中定义清晰、功能明确的组件为基于组件式的开发和大规模的软件复用提供了有力的支持，也是项目管理中计划与人员安排的依据。

Rational 公司提供了丰富的 CASE 工具支持 RUP，包括可视化建模工具 RationalRose、需求管理工具 RequisitePro、版本管理工具 ClearCase、文档生成 SoDa、测试工具 SQA 和 Perfomence 等。由于 RUP 采用标准的 UML 描述系统的模型体系结构，因此可以利用很多第三方厂家提供的产品。

在多种面向对象建模方法流派并存和相互竞争的局面中，RUP 树起了统一的旗帜，使不同厂商开发的系统模型能够基于共同的概念，使用相同的表示法，呈现彼此一致的模型风格。而且它从多种方法中吸收了大量有用(或者对一部分用户可能有用)的建模概念，使它的概念和表示法在规模上超过了以往任何一种方法，并且提供了允许用户做进一步扩展的机制。

3.2 缺点

如果软件本身并不复杂，利用统一开发软件过程提供的方法开发软件将得不偿失，反而增加了复杂性和开发成本。

不过 RUP 在取得巨大成功的同时，也不断地受到批评。来自工业界的批评主要是，它过于庞大和复杂，用户很难全面、熟练地掌握它，大多数用户实际上只使用它少部分的概念；它的许多概念含义不清，使用户感到困惑。

参考文献：

[1]Jacobson, I. Booch, G. Rumbaugh, J. [美]著，周伯生等译．统一软件开发过程[M]．机械工业出版社，2002.

[2]Sommerville, I. [英]著，程成等译．软件工程[M]．机械工业出版社，2003.

[3]Royce, W. [美]著．软件项目管理[M]．机械工业出版社，2002.

On Developing Process of Unified Software

JI Jian - hua[1], SONG He[2], YANG Yan[1]

(1. Xichang College, Xichang Sichuan 615013;

2. Xichang Animal Husbandry Bureau, Xichang Sichuan 615000)

Abstract: This paper discusses the developing process of unified software, its use case driving, framework - basis, iterative and incremental features, and also its advantases and disadvantases.

Key words: Use case driving; Framework; Iterative; Incremental