

第三代配置管理解决方案： 统一变更管理 (UCM)

一种基于活动的配置管理过程

目录

前言	3
软件开发过程中的变更	5
<i>变更是非常频繁并且是不可避免的!</i>	5
<i>管理变更的能力是项目成败的关键!</i>	5
统一变更管理	6
<i>活动和工件</i>	6
<i>贯穿生命周期的工件</i>	19
总结	23

前言

二十多年来，Rational 软件一直致力于提供全面可靠的软件开发管理解决方案，其中软件配置管理（software configuration management, SCM）解决方案集成了两个业界领先的工具：用于软件工件管理（software artifact management, SAM）的 Rational ClearCase 和用于缺陷及变更跟踪的 Rational ClearQuest。这两个工具合在一起构成了一个市场领先的软件配置管理系统，提供了真正用于加速软件开发周期和流程的解决方案，这一方案已连续四年居市场第一位（摘自 IDC 报告：“Rational’s SCM solution continued to dominate the software configuration management market in 2001, with a 36.5 percent share. Rational’s SCM solution market share grew in 2001 despite a decline in revenue in the overall SCM market during the year.”）。

在大量软件工程实践经验和用户反馈的基础上，Rational 软件提出了第三代的配置管理解决方案 - 统一变更管理（UCM）。这里，首先，让我们简单回顾一下软件配置管理的发展历程：软件配置管理在其不长的历史中不断进行演化，从七十年代基于文件（File Based）以版本控制、支持 check-out/check-in 模型和简单分支为主要特征的第一代软件配置管理，到基于项目库（Project Based）将元数据与配置项分开存储管理从而更好地支持并行开发、团队协作以及过程管理的第二代软件配置管理，发展到今天基于文件访问透明（File Transparency Based，即开发人员可以在不保留本地副本的情况下直接访问受控配置项）、全面结合变更请求管理（Software Change Management）、软件系统分析设计以及软件测试等等各个软件开发环节的完整的软件配置管理方案。而 Rational 配置管理解决方案 UCM 正是体现这一趋势的代表。

以前两代配置管理方法为基础，Rational 软件提出的统一变更管理（UCM）是用于管理软件开发过程（包括从需求到版本发布）中所有变更的“最佳实

践”流程。UCM 定义了一个可以立即用于软件开发项目的一致并基于活动的变更管理流程。通过 Rational ClearCase 和 ClearQuest 的支持，UCM 已成为 Rational 用于软件开发最佳实践的全面框架——Rational 统一过程（RUP）的关键组成部分。根据软件开发团队的具体需要，可以使用相应的过程模型来加速软件开发进度，提高软件质量并优化开发过程。

UCM 通过抽象层次的提升简化了软件开发，从而使得软件开发团队从更高的层次根据活动（activity）来管理变更。通过 UCM，一个开发活动可以自动地同其变更集（封装了所有用于实现该活动的项目工件）相关联，这样避免了管理人员手动跟踪所有文件变更。使用 UCM 还可以获得以下好处：

- 预定义的工作流程：可以直接采用预定义的 UCM 工作流程，快速提升开发组织的软件配置管理水平；
- 项目的跟踪和组织：项目管理人员可以实时掌握项目的最新动态，合理分配资源和调度开发活动；
- 协作自动化：通过将许多耗时较多的任务自动化处理，UCM 使得开发人员更多地将注意力集中在更高层次的开发活动上；
- 轻松管理基线：UCM 将开发活动嵌入到各个基线中，这样测试人员确切地知道他们将测试什么，而开发人员则确切地知道其他开发人员做了什么；
- 支持跨功能开发组：UCM 已成为 Rational Suite 产品中的核心部分，从而可以将需求到测试各个阶段的工件（例如需求文档、设计模型、应用源代码、测试用例以及 HTML 及 XML 内容等）在 UCM 框架下进行统一集成，简化了贯穿整个软件开发周期的变更过程；
- 基于同一代码构件可以进行多项目开发，简化了多项目开发管理，增大了代码共享，节省了开发资源；
- 可扩展性：小型团队可以从 ClearCase LT 和 UCM 开始，而大型团队可以结合 ClearCase 的高级构建管理（build management）功能，以及 ClearCase MultiSite 和 ClearQuest MultiSite 跨地域的使用 UCM。

本文 *第三代配置管理解决方案：一种基于活动的配置管理过程* 详细描述了统一变更管理（UCM）的概念、优点以及开发团队如何通过使用 Rational ClearCase, Rational ClearQuest 和 Rational Suite 来受益。

软件开发过程中的变更

变更是非常频繁并且是不可避免的！

今天的软件开发团队面临着巨大的挑战：一方面 Internet 驱动下的市场要求以空前的速度来开发高质量的软件应用；另一方面，软件应用需求随着开发环境和结构的日趋复杂而变得更加复杂；加上分布式开发、高性能要求、多平台、更短和连续的发布周期——这些及其他一些因素加重了软件开发一直承受的压力，实际上现在许多软件开发团队经常在能否成功开发一个新型应用上“赌博”。

由于软件开发不同于传统意义的工程技术（如建筑、机械等），市场变化以及技术上的高速更新都注定了软件变更是非常频繁并且是不可避免的，可以说变更是软件开发的基石。一方面在软件开发环境下的内部活动以新特性、新功能增强以及缺陷修复等方式不停地制造着变更；另一方面外部因素——例如新操作环境，新工具的集成，工程技术和市场条件的改善等以另一种力量驱动着变更。

管理变更的能力是项目成败的关键！

既然变更是不可避免的，那么如何管理、追踪和控制变更就显得尤为重要。尽管有多种方式可以帮助开发团队提高变更处理能力，但其中最重要的一点是整个团队的协作性，这是因为以一种可重复和可预测的方式进行高质量软件的开发需要一组开发人员相互协作。随着系统变得越来越大和越来越复杂，尽管个人生产率依然十分重要，但是决定项目成败更多的是作为一个整体的开发团队的生产率。

而软件开发团队的生产率很大程度上是由其相互协作和组织活动的能力决定的，并且开发团队的成功同其如何高效地响应不断变化的环境因素紧密相连。

对在竞争激烈的市场下想占有一席之地的开发团队而言拒绝变更无疑是行不通的，只有积极面对变更，采取有效的工具、方法和流程有机地管理、追踪和控制变更才是保证开发团队成功的关键。另外，由于各种因素的变更，原来采用的工具、方法和流程也会随着组织的成长和不停变化的需求而逐步演化，因此对软件开发团队来说另一个关键的成功因素是其扩展能力。

统一变更管理

随着开发团队的成长、产品发布周期的加速以及对软件资产（包括代码、文档等）控制的加强，对基于第三代配置管理**工具**和**过程**的需要变得越来越大。Rational 软件的统一变更管理（UCM）通过 Rational ClearCase, Rational ClearQuest 以及 Rational Suite 所提供的开发平台实现了贯穿整个软件开发周期的配置管理过程，即基于**活动**对软件**构件**和**项目**进行变更管理。

活动和工件

随着软件系统和开发团队在规模和复杂性上的不断增长，对开发团队来说如何围绕周期性的版本发布来合理地组织开发活动以及高效地管理用于实现这些活动的工件变得日益重要。**活动**（activity）可以是在现有产品中修复一个缺陷或者新加一个增强功能。**工件**（artifact）可以是在开发生命周期中涉及的任何东西，例如需求文档，源代码，设计模型或者测试脚本等。实际上软件开发过程就是软件开发团队执行**活动**并生产**工件**（图 1）。UCM 集成了由 Rational ClearQuest 提供的变更请求活动管理和 Rational ClearCase 提供的工件管理功能。

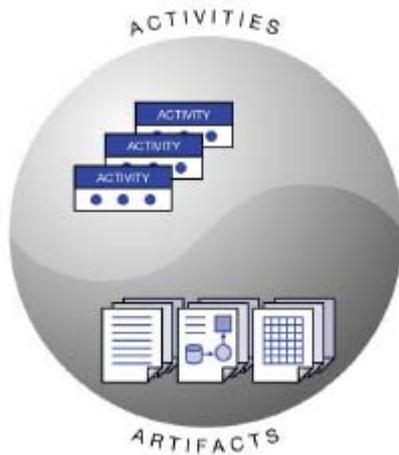


图 1: UCM 集成了 Rational ClearQuest 提供的活动管理能力和 Rational ClearCase 提供的工件管理功能

活动管理

UCM 中的活动管理是由 Rational ClearQuest 提供的，Rational ClearQuest 是一个高度灵活和可扩展的缺陷及变更跟踪系统，它可以捕获和跟踪所有类型的变更请求（例如产品缺陷、增强请求、文档变动等）。在 UCM 中这些变更均以活动出现。

Rational ClearQuest 为活动的跟踪和管理提供了可定制的工作流，这使得开发团队可以更容易地：

- 将活动分配给某个具体的开发人员
- 标识同活动相关的优先级、当前状态和其他信息（如负责人、估计工期、影响程度等）
- 自动产生查询、报告和图表

根据开发团队或开发过程需求可以灵活地调整 ClearQuest workflow 引擎：如果开发团队需要快速部署，那么也可以不进行定制，直接使用 ClearQuest 预定义的变更过程、表单和相关规则（图 2）；当开发团队需要在预定义的过程上进行定制时，可以使用 ClearQuest 对他们的变更过程的各个方面——包括缺陷和变更请求的状态转移生命周期，数据库字段，用户界面（表单）布局，报告，图表和查询等进行定制。

贯穿整个开发过程用于管理和跟踪缺陷和其他变更的一个高效 workflow 对于满足当今高质量标准及紧迫的产品工期的需要是非常重要的。UCM 提升了这些变更的抽象层次以便可以从活动的角度来观察变更，然后 Rational ClearQuest workflow 引擎将活动同相关的开发工件连接在一起。

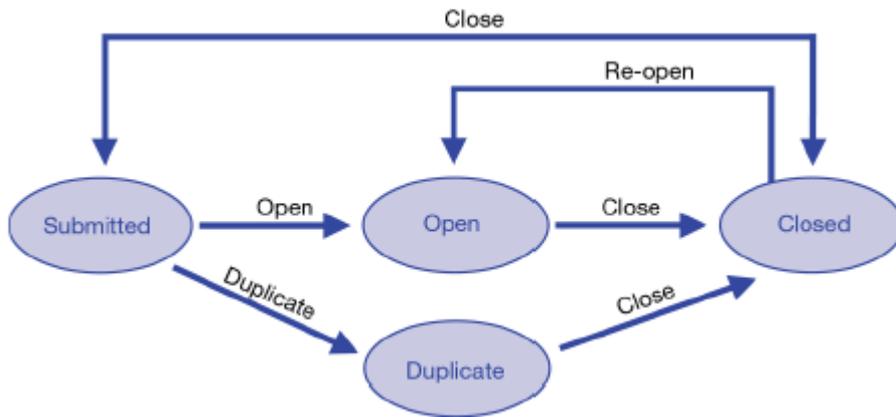


图 2: Rational ClearQuest 提供了一个现成的过程框架用于缺陷和变更跟踪 workflow

工件管理

Rational ClearCase 提供了一个软件工件管理（SAM）框架，开发团队可以使用这一框架来管理贯穿项目生命周期的所有工件。UCM 将 Rational ClearCase 基础框架同 Rational ClearQuest 中的活动管理结合在一起，从而提供了对工件和活动的集成管理。Rational ClearCase 提供了：

- 安全的工件存储和版本化

- 并行开发基础框架——无限分支能力和强大的合并功能
- 自动代码共享
- 用于选择正确工件版本的工作空间管理
- 完全的可延展性——从小型本地项目工作组到大型全球分布式开发团队

另外，Rational ClearCase 提供了灵活的 SCM 的基础框架，通过使用灵活的元数据，如标签、分支、属性、触发器（trigger）和超级链接（hyperlinks）等，开发团队可以定制他们自己的 SCM 过程。

由此可见，不同开发团队和项目可以通过 Rational ClearCase 使用不同的策略，开发团队可以从这种灵活性中受益。而 UCM 是基于一个经过验证的、成功的开发过程，因此 UCM 为希望快速启动高效 SCM 的开发团队也可以直接使用这一过程来自动实现项目策略。UCM 具体在以下六个方面提供了开发过程。

UCM: 六个过程领域

UCM 在六个具体领域提供了所定义的过程：

- 开发人员在共享及公共代码工件上的隔离和协作；
- 将一起开发、集成和发布的相关工件组按构件（component）进行组织；
- 在项目里程碑创建构件基线（baseline）并根据所建立的质量标准来提升基线；
- 将变更组织为变更集（change set）；
- 将活动管理和工件管理集成在一起；
- 按项目来组织软件开发并支持多项目之间的代码共享；

开发人员的隔离和协作

开发人员需要相互隔离的工作环境以隔离彼此的工作，避免其他组成员的变更潜在地影响其工作的稳定性。Rational ClearCase 提供了两种方式来访问工件的正确版本并在私有工作空间中在这些工件上进行工作。这两种方式是静态视图和独特的基于 MVFS 的动态视图，它们可以据本地或网络使用而分别进行实施。

静态视图为开发人员提供了在断开网络连接的情况下进行工作的灵活性，另外开发人员也可以容易地将他们的工作同开发主线进行同步。动态视图则通过一个独特的虚拟文件系统（MVFS）来实现，它使得开发人员可以透明地访问正确工件的正确版本而无需将这些工件版本复制到本地硬盘驱动器上。另外由于动态视图可以实时进行自动更新，因此紧密工作在同一分支上的开发团队无需手动更新/复制文件即可立即看到其他人员所做的变动。不管使用何种方式，开发人员都可以并行工作在多个发布版本上。例如，一个开发人员工作在发布版本 2 上，同时他也可以修复发布版本 1 中的一个缺陷，而不用担心自己的两个活动涉及的代码互相干扰或受其他开发人员的干扰。

隔离不稳定的变更对于将错误最小化是非常关键的，但是将所有的变更集成到一个所有开发团队成员均可访问的公共工作区域却是团队开发环境下的一个基本要求。今天基于构件的软件开发方法论的广泛应用以及代码变更频率和幅度的增加都要求开发团队能经常和较早地将各个开发人员的工作进行集成。以便在尽早解决可能出现的问题。

使用 Rational ClearCase，开发团队可以实现多种项目策略来同时进行工作的隔离和协作。通过强大的分支和合并功能 Rational ClearCase 可以支持大规模的并行开发。

在 ClearCase 中可以根据不同用途来建立分支，如开发人员分支，新特性分支、缺陷修复分支、新需求分支等等，从而开发团队可以根据需要建立适于自身情况的分支模型，灵活实现软件配置管理流程。但对于希望能快速利用成熟的软件开发流程的开发团队而言，UCM 则提供了一个直接可用的分支模型。实际上在 UCM 中，在分支基础上对其在更高层次上进行了抽象，从而形成了一个新概念——流（stream）。流表示一个私有或共享的工作空间，它定义了项目版本的一致配置并在 UCM 项目中的隔离和有效协作之间提供了一种平衡。熟悉 ClearCase 的读者可以将流理解为开发人员分支，UCM 中既有为每个开发人员配置的私有开发流，同时为负责集成所有交付工作的集成人员配置的公共集成流

(图 3)。由于 UCM 紧密结合了活动管理，因此其他分支用途，如特性分支、缺陷修复分支等将作为活动出现并附加到相应的工作流中。

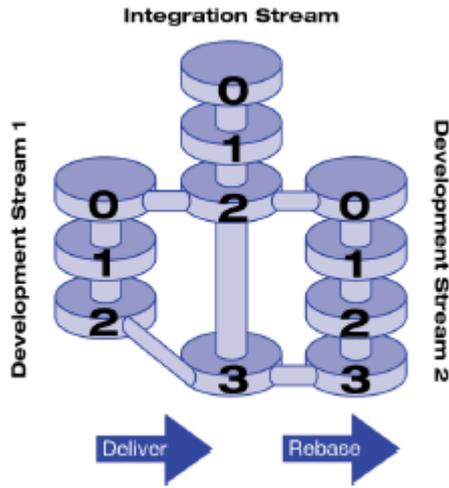


图 3：UCM 提供了公共集成工作区和私有工作区

私有开发流为开发人员提供了相互隔离的工作空间，该空间在最开始由满足一定质量标准的公共工件进行初始化。开发人员使用这些私有工作空间来进行工件的变更，构建和测试。当开发人员对他们的变更感到满意时，他们可以将这些变更交付（deliver）到公共集成流上。为了使开发人员同其他人员的进度同步，开发人员也可以用来自项目公共集成流上最新的稳定基线来变基

（rebase）他们的私有工作流。使用 UCM，开发人员可以选择什么时候进行交付和变基。

实际上项目集成流充当了所有开发人员的所有变更的协调点。为了更好地协调所有开发人员的变更集成，UCM 引入了基线（baseline）的概念作为对项目进度的度量。基线是一次构建（build）或配置的抽象表示，它实际上是构件的一个版本，而构件是相关工件的集合。项目开发团队在开发过程期间不断地创建和提升基线。随着不同开发人员交付变更给集成流，他们交付的变更将被逐一收集到项目基线中。随着基线的构建、测试和批准，它们可以被逐步提升到不同的基线级别。

基线提升级别具有两方面的功能：第一，它使项目经理或项目管理人员可以建立软件质量标准。由于当基线达到某种预定义的质量标准时就可以被标以某种基线级别，因此项目经理可以设置项目策略，标识出在哪一个基线级别（如“通过测试的”）开发人员可以执行变基操作。第二，基线提升级别就具体的开发人员应该如何同其所开发的工件进行交互提供了指导。例如，根据某条基线通过某些冒烟测试的时间可以帮助测试人员确定什么时候开始测试。

构件基线

第二个 UCM 过程领域是将工件组织为构件。在第二代配置管理中，大多以文件版本形式来管理所有的文件，当一个复杂项目中包含成千个文件上万个版本时，整个项目的开发控制将变得相当复杂，因此对众多的文件进行合理分类以呈现系统的设计要素可以大大简化项目开发控制。

UCM 通过将多个工件组织为构件（在 UCM 中构件指一个 VOB 的根目录或 VOB 的某个第一层子目录）从而扩展了软件工件管理的版本控制能力，并且 UCM 还提供了用于自动化构件管理的工具和过程。即用基线对构件而不是构件中众多的版本进行标识，然后用这一基线作为新的开发起点并更新开发人员的工作空间。

构件基线是在 ClearCase 标签（label）的基础上结合活动管理所做的扩展，即您可以知道一个 UCM 构件基线中包含了哪些开发活动，例如一条基线可能包含了三个开发活动，如 BUG 101 的修复、用户登录界面汉化以及新增打印特性的支持。

对于包含多个构件的复杂系统，UCM 提供了基于多个构件的组合基线，即多个构件之间可以建立依赖关系，一旦底层构件的基线发生变化，例如生成一条新基线，其上层构件相应地也自动建立起一条基线，该基线自动包含底层构件基线。例如，一个较为复杂的 MIS 系统包含“数据库访问”，“业务逻辑处理”和“前端图形界面”三个构件，其中“前端图形界面”构件依赖于“业务逻辑



处理”构件，而“业务逻辑处理”构件依赖于“数据库访问”构件。这样当“数据库访问”构件发生了变化并新建了一条新基线（如 DB_BASELINE_Dec24）后，在“业务逻辑处理”构件和“前端图形界面”构件各自自动建立了一条新基线（如 BUSINESS_BASELINE_Dec24 和 GUI_BASELINE_Dec24）。这样上层构件的最新基线可以自动跟踪底层构件的最新基线。

构件管理的自动化对于高效无误地开发可能包含数千个源代码工件（还有其他相关的工件，如 web 内容，设计模型，需求说明和测试脚本等）的复杂软件系统而言意义重大。

构件基线提升

项目开发团队的成员工作在一个 UCM 项目（project）中，项目经理通过配置软件构件从而使项目成为由构件构成的体系结构。大多数组织将 UCM 管理的构件设计为可以反映出他们软件体系结构的方式（图 4），即将所有相关工作件按体系结构组织为有意义的子系统，进而放入不同的构件中。

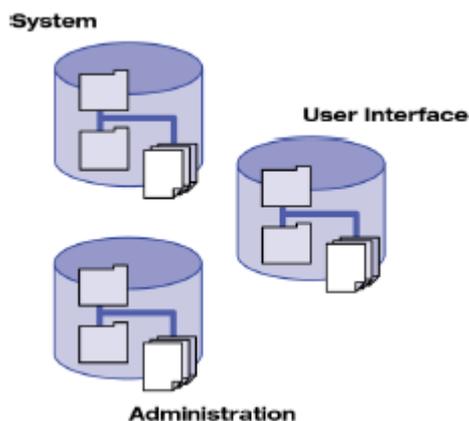


图 4：用 UCM 构件直接对软件体系结构建模

如上节所描述的，开发人员在交付变更到公共集成流时可以周期性地更新他们私有开发流中的构件。然后开发团队可以根据开发过程的当前阶段和质量级别对构件进行评级。项目策略确定了在开发人员变基之前构件基线必须达到的质量级别以及其他开发团队成员（如测试人员）应该如何同构件基线交互。在稍后会对项目以及项目策略做更多描述。

UCM 提供了五种预定义的基线级别，它们包括被拒绝（rejected），初始（initial），通过构建（built），通过测试（tested）和已发布（released）。另外，UCM 允许开发团队用他们自己的命名规范和提升策略对这些预定义基线级别进行定制。

变更集

第四个 UCM 过程领域是将独立的工作变更组织为可作为整体进行交付、跟踪和管理的变更集中。由于通常当开发人员工作在一个活动（例如缺陷修复）上时，他们很少只修改一个文件，因此用变更集可以表示用以完成某个具体活动的工件的所有变更，例如为修改一个编号为 63 的缺陷变动了 50 个目录/文件的 120 个版本，则缺陷 63 的变更集为相关的 120 个文件/目录版本。当开发人员同时工作在多个变更请求上的需要使得这一过程更加复杂。例如，一个开发人员在进一个新发布版本的开发，这时由于当前发布版本的一个错误要求他不得不中断当前的开发工作转而去修复这一缺陷，这样该开发人员必须在同一工件上进行两种不同的变更，一种是在未来发布版本中的增强功能变更，另一种是在前一发布版本中修复缺陷的变更。

通过将同一个开发活动相关的所有变更收集到一个变更集中，UCM 简化了管理多个工件变更或者多个工件版本的过程。UCM 围绕具体的开发活动来进行工作组织，同时 UCM 还确保已完成的活动包含所有必要工件上发生的所有变更。

活动和工件管理

第五个 UCM 过程领域是通过使用一个可将活动及其相关工件集链接起来的自动化 workflow（图 5）将活动管理和工件管理集成起来。这给了开发团队极大的灵

活性来为不同类型活动的管理指定不同的 workflow。UCM 提供了最常用活动类型的预定义 workflow，包括缺陷修复和增强请求。

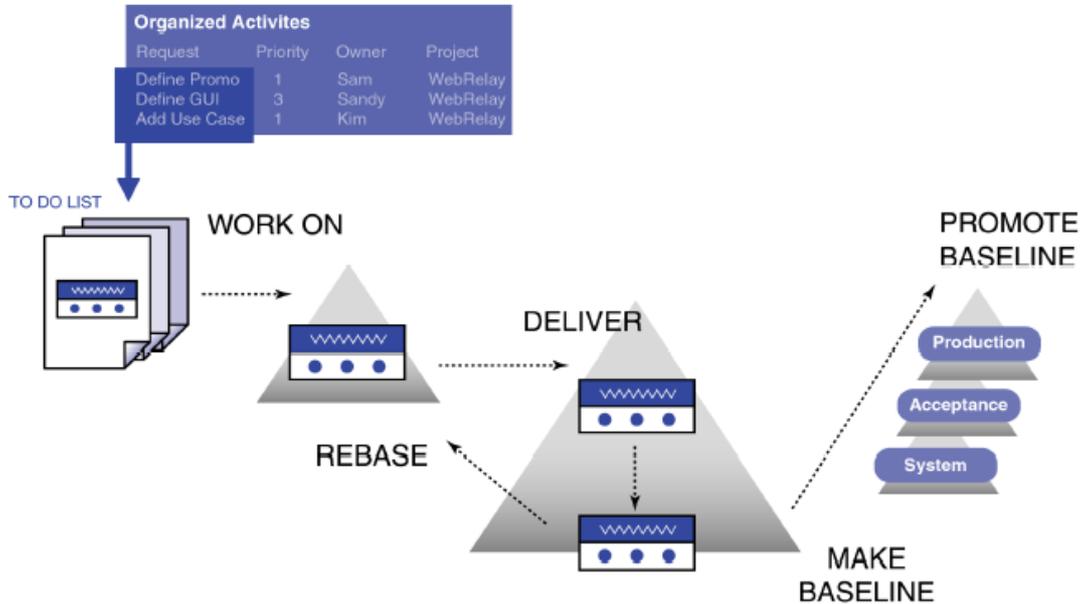


图 5: UCM workflow 概览

开发团队还可以使用 ClearQuest Designer 这一模块来对这些预定义过程进行定制，项目经理或者项目管理人员可以用它来创建所有需要的活动类型，包括缺陷修复，增强功能请求，文档变动以及 Web 网站变动等。使用 ClearQuest Designer 的图形用户界面，项目经理也可以定义字段，表单以及每个记录类型的状态转移。

为了方便开发人员更容易地标识活动和项目代码库中工件间的关系，UCM 自动将活动和其相关的变更集链接起来（图 6）。当在一个 UCM 项目中工作时，开发人员所交付的是活动（见开发人员隔离和协作一节）而不是文件形式的工件。类似的，当开发人员变基时，他们根据新构件基线提供的活动（而不是文件形式的工件）来重审将要在他们的私有开发流中接收的变更内容。这样，开发人员不仅可以看到所有相关工件完整的版本历史，而且可以看到实现每个变更的所有活动。这给了开发团队一个项目是如何从一个阶段演进到下一个阶段

演进的全面视图，当需要标识出一个工件版本的变更是如何影响另一个版本时，这一优点所带来的价值是无法估计的。

使用 UCM 一致、可重复的用于活动管理的过程，通过活动管理和工件管理的集成可以帮助开发团队减少错误，开发人员还可以避免许多通常需要手工作业的单调工作，从而更有效率地完成开发任务。

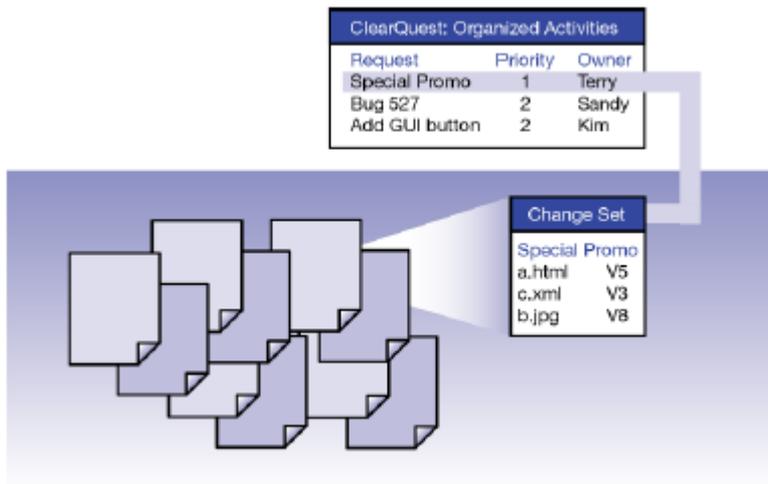


图 6: UCM 将活动和相关的工件链接起来

项目和项目策略

UCM 中的项目可以和实际软件开发中的各种项目对应，每个 UCM 项目包含一个集成工作流和若干个私有开发流，项目可由一组构件构成。这里需要强调的是，一个构件可以被多个项目共享，进而项目 A 中的开发人员可以对一个构件进行修改，创建新的构件基线；而项目 B 的开发人员可以参照同一构件以及该构件在项目 A 中生成的基线，但不能对该构件进行任何改动。因而 UCM 项目从代码级提供了软件共享及重用的良好基础。例如某系统集成公司的湖北分公司刚刚结束了为某银行开发的客服系统 1.0 的开发（我们可以将最终交付的版本称之为 XBANK_CRM_HuBei_1.0），与此同时该系统集成公司的北京分公司正准备为另一家银行进行类似系统的开发。另外 XBANK_CRM_HuBei_1.0 在推广过程中出现了一些问题，亟待解决。还有该客服系统 2.0 版 XBANK_CRM_HuBei_2.0 的需

需求分析准备在下月开始。此时如果利用 UCM 的多项目机制则可以为在北京的开发团队建立一个新项目 CRM_Beijing_1.0, 但该项目并不是从零开始的, 它可以从 XBANK_CRM_HuBei_1.0 开始, 有选择地借鉴 XBANK_CRM_HuBei_1.0 中可共享的部分。同时原有的 XBANK_CRM_HuBei_1.0 的开发人员可以继续进行 2.0 版的研制工作。而为了解决 XBANK_CRM_HuBei_1.0 中在推广中出现的问题又可以新建一个 XBANK_CRM_HuBei_1.0 版的维护项目 XBANK_CRM_HuBei_1.0_BUGFIX, 部分原有开发人员既可以加入该项目进行原有 1.0 版的错误修复, 又可以互不干扰地在原有项目中进行 2.0 版的需求分析。

UCM 在项目上另一个突出特点是不同项目中的开发流之间可以进行跨项目的工作交付。即一个项目可以将一条构件基线中包含的工件交付给另一个项目, 也可以将一个开发流上的活动变更集交付给另一个项目。紧接上例, 假设过了三个月后项目 XBANK_CRM_HuBei_1.0_BUGFIX 修复了 8 个缺陷, 而项目 CRM_Beijing_1.0 也根据北京用户的特点实现了 3 个不同的新特性, 这时如果需要可以在恰当时机将这两个项目中的成果有选择地交付给正在湖北进行的客服 2.0 版的开发, 从而使得 2.0 版不仅修复了前一版本中的错误, 而且具备了 3 个新特性。

可以看出项目从一个更高层次上支持了传统意义上基于分支的并行开发, 这对于软件开发组织从面向项目到面向产品的转变, 合理利用软件开发人力资源, 改善软件产品体系结构从而支持更为复杂的软件开发具有重大意义。

由于同一代码构件上多项目开发的引入, 相应地 UCM 加入了多种项目策略来进行支持, 用户可以根据需要灵活定义项目策略。例如上面提到的基线级别, 可以定义只有基线级别提升到“Tested”才能作为其他开发工作流的推荐基线。另外, 对于多项目间的代码交付和共享, 还可以定义是否允许接受其他项目的变更等。除了项目一级的策略以外, 在工作流一级 UCM 也有类似的访问策略, 如某个工作流是否允许接受来自本项目或其他项目的工作流上的变更, 是否允许向本项目或其他项目的工作流交付变更等等。

成功的软件配置管理需要同时重视人的活动和得到整个开发团队的全体接受。

如果开发团队成员不能或不采用 SCM 方法论的话，SCM 实现将会失败。

Rational 软件多年来从成功的软件开发团队和自身的软件开发中吸取最佳实践经验，将 UCM 作为一个可为开发团队采用的优化过程来进行设计。UCM 的功能已全部嵌入到 Rational 统一开发过程管理（Rational Unified Process, RUP）中，RUP 是在软件开发的各个生命周期应用最佳软件开发经验的一个实用框架。软件开发团队通过直接访问 RUP 并通过 Rational ClearCase, Rational ClearQuest 和 Rational Suite 的工具专家可以得到关于 UCM 过程及其详细描述。

UCM 通过使开发人员可以完成下面的工作而简化了他们的工作：

- 快速准确地访问正确工件的正确版本
- 在他们的工作空间中相互隔离地进行各自的工作，但可以选择什么时候交付自己的工作及接收其他人员的变更
- 使用项目策略来标识什么时候质量达到了可以接受的级别从而可以在自己的私有工作空间中接收其他人员的变更
- 从他们自己的桌面透明地访问 SCM 工具。使用可同他们最常使用的工具（如 Visual C++）进行集成的工具来进行他们的工作——在开发活动上没有附加不适当约束或增加工作时间
- 自动而透明地将变更集同活动进行集成
- 自动进行活动的跟踪、管理和报告。

另外，UCM 还减轻了项目经理的负担，从而项目经理可以

- 通过一个预定义的生命周期 workflow 在任何时间标识一个具体活动的状态
- 用统一的报告来监控最新的项目状态
- 将精力集中在高优先级或紧急的事件（活动）上
- 确定工作负载并平衡任务分配

贯穿生命周期的工件

到现在为止，本文主要集中在源代码和其相关工件上，如对象文件和头文件等。但是使用 UCM 时贯穿生命周期的变更也可以由非开发人员进行管理，这样就最佳实现了统一变更管理的全部好处。这些非开发人员包括分析人员，设计人员以及测试人员（图 7）。相应的工件包括他们在相关领域产生的工件，例如分析人员所创建的需求文档和用例（use cases），设计人员所建立的设计模型和用例，测试人员建立的测试脚本，测试数据和测试结果。

为了高效地通信和协作工作，开发团队成员需要有效地共享这些工件。

Rational Suite 包含有一个集成平台，通过这一公共平台可以访问贯穿多个领域的所有类型的开发工件。作为 Rational Suite 的一部分，UCM 提供了一个用于管理贯穿软件开发生命周期的信息共享的过程层。现在每个 Rational Suite 产品套件中均包含了这两个产品，这样每个 Rational Suite 产品都包含了对 UCM 过程的支持。非软件开发人员（如分析人员、设计人员以及测试人员）可以应用 UCM 原理象控制代码变更一样来控制他们生产的工件（如需求文档，用例，设计模型，测试脚本及测试数据）。Rational ClearQuest workflow 引擎强化了活动管理，另外一致的工作流使得所有的团队成员都可以容易地标识优先级，而 ClearQuest 提供的工作列表（to do list）特性可以使每个人均可从他们的桌面透明访问待进行的工作（活动）列表，从而容易地标识出他们下一步需要进行的工作。同样构件基线是新工作（分析、设计、开发及测试）的基础并指导团队成员什么时候更新他们工作空间。

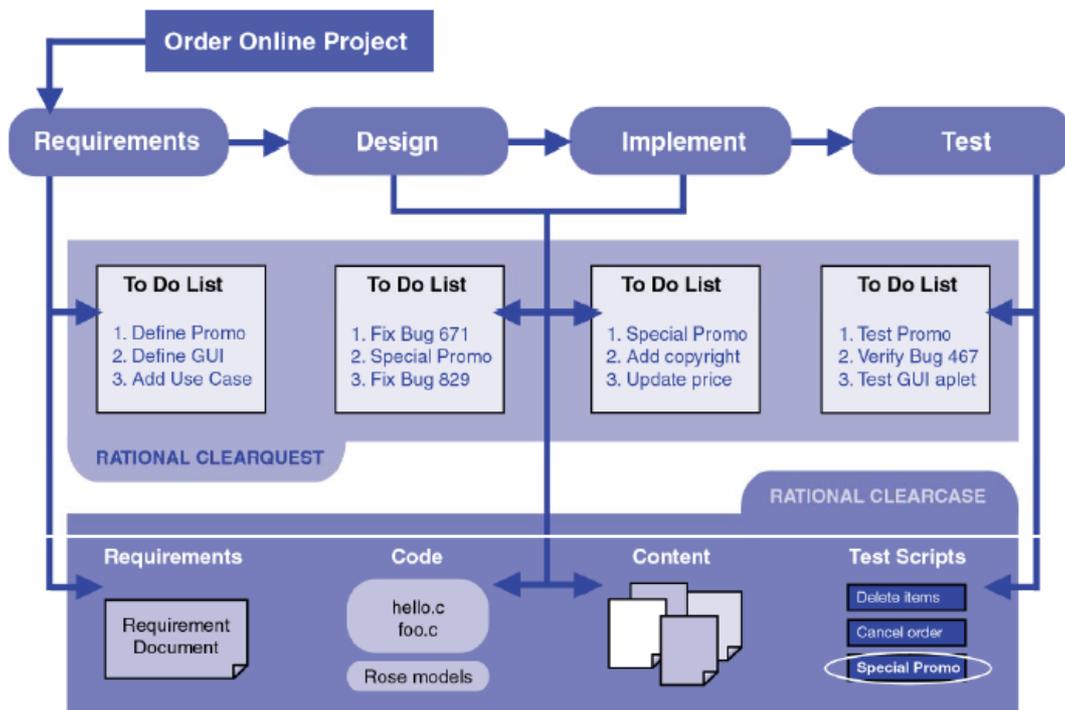


图 7：开发团队成员在生命周期的不同阶段产生不同的工件

来自分析的工件

分析人员扮演着定义项目范围的重要角色，分析人员需要确定解决方案是什么以及系统边界。在分析期间这些专业人员创建了多种不同的工件来帮助解释说明所建议的解决方案，这些工件包括需求文档，用例以及可视化模型。开发团队在整个开发过程中不断使用这些工件来管理项目变更是必不可少的。

成功的项目依赖于正确的需求管理。高效的需求管理包括减少开发的进度风险和系统的不稳定性，以及跟踪需求变更。项目管理，风险评估以及相关评估标准的产生都依赖于需求管理和需求的可追溯性——即开发团队以一种规范过程接受变更并审查需求变更的历史。

Rational RequisitePro 是 Rational Suite 开发团队统一平台 (Team Unifying Platform) 中用于需求管理的工具。软件开发团队可以使用 RequisitePro 来创建、管理和跟踪分析工件——例如需求属性、需求说明和管理计划，以及用例模型，术语表以及涉众 (stakeholder) 请求。

UCM 项目以同管理代码相似的方式管理需求，此外 UCM 还将这些需求工件包含在构件基线中。这样分析人员不仅知道哪些活动构成了一条构件基线中的工件，还知道哪些需求指导着这些工件的开发。

来自设计的工件

设计人员对系统基础结构建模并使用用例和设计模型进一步细化系统定义。通过设计工件对系统进行抽象，可以减少整个系统的复杂性。

在实际开发过程中经常会出现这样一种情况：一些开发团队不能继续使用一些设计模型，因为正式的编码工作已经开始。这是一个错误，因为这些设计模型在帮助规划整个工作，另外对于协助新的组成员快速上手，度量正在进行中的变更请求的影响，以及评估整个项目风险都非常有价值。由 Rational Rose 提供的双向工程功能可以帮助开发团队保持当前编码进度和这些模型之间的同步（图 8），而 UCM 确保模型与代码是最新的。

在 UCM 项目中，构件基线包含模型工件和代码工件。Rose Model Integrator 和 ClearCase 比较/合并特性的紧密集成使得开发团队可以尽可能方便地在设计模型的同时进行编码。这时 UCM 中的变更集（参见变更集部分）可以扩展到模型元素，这样变更集便包含了代码工件上的变更以及模型工件上的变更。

Rational Rose 支持 UCM 交付（deliver）和变基（rebase）操作（参见活动和工件管理部分），使得多个设计人员可以在 Rose 模型上同时工作，如同开发人员并发工作在代码上一样。

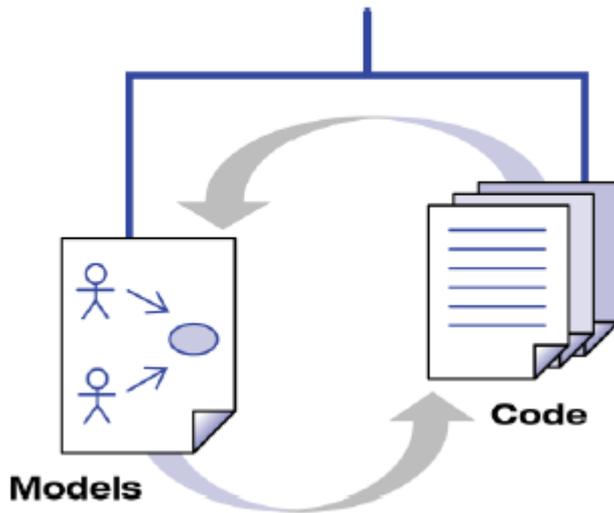


图 8：并行开发过程中模型和代码之间的交互

来自测试的工件

传统的开发方法将测试工作放到开发的末期，在编码工作之后进行。成功的构件开发团队都体会到单元测试（unit test）对于项目成功是非常重要的，使用单元测试就是不将构件的质量保证放到集成阶段，而是预先对单元、子系统和系统进行测试。

所有这些并发测试产生了大量的测试工件——包括测试需求、测试脚本、测试数据和大量的测试结果。Rational Suite TestStudio 以及 Rational QualityArchitect 为开发团队提供了可持续地生产高质量软件工件所需的集成框架。借助 UCM，开发团队可以并发地管理他们的测试工件和相关的开发工件。

今天，大多数开发团队都意识到将这些测试工件和构件基线集成到一起的好处。在 UCM 项目中，可以在一条构件基线中包含所有的代码工件以及相关的测试需求，测试程序和测试数据。

这样前面描述的变更集就可以扩展到测试用例，测试脚本和测试数据。集成的活动和工件管理将活动的整个范围——从对代码变更的测试校验活动——到由其活动描述的 UCM 变更集链接起来。

分析，设计，编码和测试工件

Rational Suite 和 UCM 提供的优点是无价的。构件基线在一条基线中包容了所有的项目工件——从需求文档到测试包——这对于任何需要进行维护和升级任务的开发团队都是一个大优点。借助 UCM 基线级别，使用 Rational Suite 跨功能的多个组可以更容易地：

- 标识所需的活动
- 确定什么时候变基
- 标识需测试的活动
- 评估开发中的工件

另外，UCM 基线级别帮助开发团队识别什么时候开始跨功能的活动，例如，在编码结束后经过一系列单元测试和冒烟测试（smoke test）一条基线达到了可以进行集成测试的质量级别，这时可以提升该构件基线从而启动集成测试活动。

在 UCM 中，质量监控是依靠 Rational ClearQuest 提供的快速报告功能来进一步进行加强的。通过这些报告功能，组成员可以生成清晰而简明的项目状态数据以便快速探测问题，评估风险并快速达成解决方案。另外项目管理人员可以将工作围绕高优先级的活动展开并且确保所有的发布版本都满足预定义的质量级别。

总结

随着 Internet 持续增长并逐步成为同客户交易和收集战略信息的主要渠道，软件开发团队建立支持这些功能应用的压力也在增长。这一趋势使得软件开发团队（包括所有类型 and 所有规模的软件开发团队）对用于管理变更的、可预测和



可重复的过程的需要变得日趋迫切。通过 Rational ClearCase, Rational ClearQuest 以及 Rational Suite 中的统一变更管理能力, Rational 软件提供了用于控制软件开发中变更的全面软件配置管理方案。基于多年来对软件开发领域的观察和 Rational 软件开发而得到的最佳经验, Rational UCM 通过活动来组织变更并通过构件和项目在更高的层次上抽象软件开发, 从而提供了一种实用的解决方案来解决 Internet 时代高质量软件应用带来的挑战。