一个基于软件配置管理的过程管理框架*

彭　鑫，赵文耘，张　志

（复旦大学计算机科学与工程系软件工程实验室，上海　200433）

E-mail：cspengxin@163.com；wyzhao@fudan.edu.cn；zhangzhifd@163.com

摘　要：在软件质量体系的诸多支持活动中，配置管理处于核心地位。基于配置管理进行过程控制可以提高过程管理的有效性。引入过程模式的思想，在此基础上提出了一个基于配置管理的过程管理框架。这个框架以各种抽象层次上的子系统和构件为过程管理单位，在过程模型层、小组协作层和个人活动层三个层次上进行过程控制，并分别采取不同的过程控制策略，提高了配置管理和过程管理的有效性与灵活性。

关键词：配置管理；软件过程；过程管理；过程模式；变更管理

随着软件系统规模和复杂性的不断增大，软件开发过程也越来越复杂，越来越需要通过一些手段来加以控制和规范。软件过程控制和软件配置管理都是提高开发效率，保证软件产品质量的重要手段。软件过程控制通过定义、描述并执行软件过程模型来提高软件开发过程的质量。而软件配置管理主要关注于记录并控制软件开发过程所产生的全部资源的演化[1]。传统的配置管理主要包括版本管理、变更控制、状态统计、审计和评审等。文献[2]将过程支持和团队支持也纳入到软件配置管理功能之中。

过程建模领域已经有许多有价值的研究成果，例如基于产品流的过程建模[3]、基于过程模式的过程框架[4]。这两种过程模型都是描述性的。也有一些研究关注于可执行的过程模型，例如基于工作流的过程建模。但总的来说这些过程模型对于实际的软件开发过程控制力有限。这方面的困难主要是由于缺乏一种将过程控制与软件开发活动有机结合起来的手段。

在质量体系的诸多支持活动中，配置管理处于支持活动的中心位置，它有机地把其它支持活动结合起来，形成一个整体，相互促进，相互影响，有力地保证了质量体系的实施[5]。在一个全面实施了软件配置管理的机构中，开发者、项目管理者等都将以配置管理系统为核心开展软件工程活动。因此，在配置管理的基础上进行过程控制可以使得过程管理的可操作性更强。另一方面，配置管理本身也需要软件过程管理的支持：版本管理中配置项的演化过程必须符合相应的过程模型；变更管理过程需要控制变化实施的全过程并追踪所有变更的相关信息[6]。通过过程控制，各种软件资源可以建立起系统的过程追踪关系，从而为缺陷追踪和变更影响分析等打下良好的基础。

基于这种观点，本文结合过程模式的思想，提出了一个基于配置管理的过程管理框架。这个框架以子系统和构件为基本的过程控制单位，按照每次迭代的目标将过程步的执行分解为活动。活动将按照一定的过程模式执行。过程模式库为各种活动的执行提供可选的方案。通过不同环境下模式的灵活运用，过程控制表现出高度的灵活性。过程控制将依托配置管理进行，因此过程模式的描述和执行将基于配置项及其演化规则。变更管理与过程控制也在此框架下统一起来，形成完整的变更控制和变更追踪体系。在此基础上，我们实现了一个配置管理工具原型 FDSCM。

1　概述

1.1　过程管理的新需求

现代的软件开发过程面临着越来越多的不确定因素，包括需求、技术、商业目标等。而软件项目中涉及的软件资源也多种多样，包括各种各样的文档、源代码、图片、音频/视频文件等，它们的开发过程有很大差别。这些都要求软件过程应该具有足够的灵活性，以适应不同的情况。

在传统的软件开发过程(例如瀑布模型)中，过程控制往往以整个项目为单位，贯穿整个项目生命周期。这种开发方式规定了一些过程活动，却没有指明如何去执行。当开发活动细化到一定程度时，过程模型已经无法提供所需的过程支持。而且，这种过程控制方式的灵活性不足，无法适应需求、技术等的频繁变动。而在基于复用的软件开发过程中，项目组通过构件集成组装得到最终系统[7]，十分强调项目间的资源复用。构件的来源多种多样，包括企业级的通用构件、项目特定构件以及一部分第三方构件。所面临的开发方式也各不相同，既有直接编码的项目模块和原子构件开发，又有组装式的复合构件开发。这就要求过程控制能够适应不同的开发方式，提供灵活多样的过程管理方案。

基于以上这些分析，我们认为过程控制应该具有较高的灵活性，能够适应不同的开发过程。构件应该单独成为一类过程控

---

* **基金项目**：国家 863 高科技发展计划资助项目(2001AA113070)；上海市科学技术委员会科技攻关项目(035115026)

作者简介：彭鑫(1979-)，男，湖北黄冈人，博士生，主要研究领域为软件工程，企业应用集成；赵文耘(1964-)，男，江苏常熟人，教授，博士生导师，主要研究领域为软件工程，电子商务，企业应用集成；张志(1976-)，男，河北石家庄人，硕士，助理工程师，主要研究领域为软件工程。

制的单位，相应的过程管理可以超出项目的范围进行。同时过程控制应该分层实施，在不同的层次上采用不同的策略。

1.2 关于 FDSCM

FDSCM 是一个基于构件的配置管理工具原型，它的一个显著特点是支持构件的独立演化以及分层的开发视图。FDSCM 支持两级版本管理：服务器级以配置项、构件和子系统为配置管理单位，基于过程模型和模式提供过程支持；每个开发者都拥有自己的本地工作区，在本地级上执行以文件和配置项为单位的本地版本管理，同时以活动为单位进行过程控制。这些特性使得我们的过程管理框架能够得到较好的支持。

2 基于配置管理的过程管理框架

2.1 配置项、子系统与构件

文献[1]提出了一种基于子系统的配置管理框架。在这个框架中，子系统为配置项提供了一种层次组织，同时也是变更控制和变更追踪的基本单位。我们的过程框架同样引入子系统作为配置项的组织单位，同时也作为过程控制的单位。同一项目中的各子系统采用相同的过程模型。构件是另一种过程控制单位。构件的开发方式与子系统不同，可以在项目间复用和演化。因此每个构件都可以有自己的过程模型，当构件在不同的项目中演化时都将遵循该过程模型。图 1 描述了子系统和构件的结构，相关的概念解释如下：

图 1 子系统和构件的组成结构

- 配置项：原子配置项由文件和目录组成，复合配置项由其它配置项按照一定的结构复合而成。FDSCM 中的配置项分为过程配置项和非过程配置项。过程配置项具有过程属性，属于过程模型中的某个步骤，而非过程配置项不具有过程属性。
- 子系统：由构件、配置项和其它子系统组成，是项目的基本组成单位(项目本身也可视为子系统)，同时也是过程控制的基本单位。
- 构件：由配置项组成和其它构件组成，分为原子构件和复合构件[8]，可以独立于项目演化，因此具有自己的过程模型。

配置项是配置管理中最小的逻辑单位和版本实体。子系统和构件是执行过程模型的基本单位。这里的子系统并不完全等同于一般的系统模块的概念。实际的项目开发过程中，一个独立执行过程模型的部分就应该划分为一个子系统。构件在这里并不是指源代码构件或者二进制构件，而是包含了一个构件的整个生命周期，例如构件需求文档、设计文档、源代码以及编译后的二进制构件等。

2.2 过程控制的三个层次

在项目的开发过程中，既应该有统一控制的部分，这部分通常粒度较高，变化也较少；又应该允许在一定范围内根据实际情况选择过程方案，这部分粒度较低，同时具有较大的不确定性。例如，项目开发采用瀑布模型，按照“分析、设计、编码…”这样的过程步骤进行统一控制。但这些步骤的具体执行过程就有很大的不确定性了，例如：普通模块和构件的设计过程不一样；同一份设计文档可能既包含自然语言部分又包含形式化描述部分，它们各自的开发过程又不一样。

从另一个角度看，过程控制既存在于小组协作中，又存在于个人开发过程中。这两个过程层次上的软件资源具有不同的“可见度”和配置管理策略。例如在“子系统编码”这样一个小组协作过程中，一位开发者分配到的是某一个类的代码编写工作，那么他可能要遵守“编码—代码复审—单元测试”这样的过程。这个过程属于个人开发过程，对于小组协作不可见(小组协作过程只关心他交出了合格的类代码文件)。但实施这种过程控制仍然有意义，它可以使得个人开发活动的可管理性更强，质量更有保障。注意，这里的个人开发过程是指开发者在完成具体的个人开发任务时所遵循的过程，不同于用于改善个人工作方式的个人软件过程(PSP)[9]。

由此可见，过程控制在不同层次上体现出不同的特点和要求，因此必须特别对待，并采取不同的策略。首先，我们定义过程步骤、活动和过程模式如下：

- 过程迭代：为达到特定目标而进行的过程模型的一次完整或部分执行。
- 过程步骤：过程模型中定义的开发步骤，在每次过程迭代中的具体目标不同，所开展的活动也不相同。

• 活动：为完成过程迭代中特定过程步骤的演化目标而开展的开发活动，配置项的创建及演化将在活动中进行。

• 过程模式：为活动提供执行方案，通过定义子活动序列对活动进行分解。

我们的过程管理框架包含下面三个层次：

1)过程模型层：这个层次的过程控制按照特定的过程模型在各子系统和构件上开展，主要支持过程模型的迭代执行。这个层面上的过程控制主要体现软件项目的整体过程控制，以推动各子系统/构件的版本演进为目的(项目是最高层次的子系统)。过程模型中各步骤的每次执行都可能涉及一系列的小组协作层和个人活动层过程。

2)小组协作层：过程步骤的执行将以小组协作的方式开展。活动本身只描述作什么，而过程模式则为活动提供执行方案[4]。过程模式本身包含一系列的活动及其执行顺序，因此过程模式将所执行的活动分解为一些子活动。这种基于模式的活动分解可能会进行多次。活动执行的结果体现为一些配置项的产生或版本演化。

3)个人活动层：小组协作过程最终将分解为个人开发过程。这种过程以完成小组协作过程中的一个环节为目标。相关的过程控制信息在个人工作区上维护，对外仅需提交满足一定要求的软件制品。

这样的三个层次上体现出不同的过程控制和配置管理策略。过程模型层体现了项目本身的特点，在项目开发过程中将保持稳定，保证软件系统按照一定的规程进行演化。而小组协作层和个人活动层上的过程控制都是面向具体问题，以完成某个活动为目标。个人活动层是原子态的过程控制，是构成其它两个过程控制层次的基础。

2.3　过程管理框架

根据这样的三个过程控制层次，我们提出了一个过程管理框架(图 2)。在这个框架中，不同的层次上执行不同的过程控制策略，控制与之相应的一部分配置管理操作。

图 2　过程管理框架

过程模型层体现了特定项目的总体过程管理方案。过程模型描述了一个软件系统的基本开发路线，例如需求分析、设计、编码、测试等基本环节。基于 2. 1 节的分析，我们认为同一项目中的子系统应该执行相同的过程模型，而每个构件则拥有自己的过程模型。在这个层次上，迭代是过程控制的主线，每一次迭代都体现了一定的演化目标。例如如果采用包含上述四步骤(需求分析、设计、编码、测试)的开发模型，那么可能存在为开发一个测试版本而进行的一次完整迭代和为修正某个设计缺陷而从设计步骤开始的一次迭代。过程模型可以包含分支，但每一次迭代都将遵循多种可能的开发路径中的某一条。

小组协作层上进行的主要是活动的分解和执行过程。从过程模型层的过程步骤开始，按照本次过程迭代中该步骤的目标确定需要进行的活动。这些活动将按照一定的过程模式进行。过程模式定义了下一级抽象层次上的活动序列及相应的演化策略。随着活动的层层分解，一系列配置项将建立或发生演化。

当开发活动分配到单个开发者后，过程控制将进入个人活动层。这个层次与小组协作层一样基于过程模式开展，所不同的是相关活动将基于文件和配置项的本地版本管理执行。这个层次上的过程模式和过程活动具有较低的粒度，因此本地活动一般不再进行分解。

当相关活动结束后，过程步骤以过程集成的方式结束本次迭代的这个步骤。在 FDSCM 中，子系统或构件的每个过程步骤都对应于一个特殊的配置项，我们称之为过程配置项。过程配置项的每个版本都代表了该过程步骤下所包含的配置项的一个里程碑配置。每次过程集成都将产生所对应的过程配置项的一个新版本。然后，过程模型中的后继步骤依次展开，直至完成本次迭代。由此可见，过程模型层是提出问题的层次。过程步骤的目标决定了将要进行的一系列配置管理活动。小组协作层和个人活动层体现了在此目标下任务的层层分解。个人活动层体现了活动分解的原子态，将在配置工具的个人工作区中执行。

3 基于过程模型的生命周期管理

3.1 过程集成

在 FDSCM 中，每个子系统或构件的开发都将遵循相应的过程模型。模型中的每一个过程步骤都对应一个复合配置项，例如一个子系统的设计步骤集成代表了该子系统的完整设计。这个复合配置项就是过程配置项。过程模型层的每次迭代中，每个过程步骤结束后将以过程集成的方式产生此过程的新版本。过程集成将属于某个过程步骤的过程配置项进行复合，从而产生新的过程配置项版本。由于子系统可能包含下一级子系统，因此参与过程集成的还有下一级子系统相同过程步骤的集成配置项。一次迭代过程中所涉及的各步骤过程配置项版本的组合代表了该子系统的一个完整的迭代版本。

构件的开发视图和复用视图应该加以区分[10]。构件由于其封装性，通常只会暴露有限的接口。例如代码构件可能只在编码阶段参与系统构造，此时该构件的内部设计是不可见的。因此构件的过程控制只在构件本身的开发和维护中进行，当参与系统构造时将只属于特定子系统的某个过程步骤。

3.2 过程生命周期

图 3 描述了过程模型层的生命周期控制，其中每个方框代表一个过程配置项版本。图中横向箭头表示一次迭代中各步骤之间的追踪关系，纵向箭头表示过程配置项版本的演化关系。从图中可以看出迭代可以从中间步骤开始，例如从设计阶段的 1。1 版本开始的一次迭代。这种情况下生命周期仍然是完整的，因为这意味着此前的过程步骤上不需要进行演化，这一部分可以从前一次迭代继承。

过程模型层是过程控制的最顶层，相应的过程管理策略是基于过程模型的迭代控制。在我们的过程框架中，每一次迭代都必须面向特定的目标。根据目标的不同，迭代既可以由起点步骤开始也可以由某个中间步骤开始。每次迭代都将以某个终点步骤作为结束。

由此可见，参与过程迭代的过程步骤或者是本次迭代的起点，或者由某一前驱步骤所引发。例如在图 3 中的过程模型中，分析阶段产生新的分析过程配置项版本后，设计阶段必须按照新的分析结果对相关的设计进行修改，相应产生新的设计版本。同样，编码和测试步骤也必须依次进行以完成本次迭代。每一过程步骤都将根据一定的目标进行。迭代的起点步骤目标将取决于迭代的整体目标。例如，以提高子系统性能为目标的一次过程迭代将从设计阶段开始，因此提高性能成为设计步骤的目标。后续步骤的目标均由前一步骤决定。例如，新的设计可能对某一算法作出了改进，相应的编码阶段就要对相关代码进行修改。由此可见，在过程迭代中，每个过程步骤以特定目标开始，以过程集成结束。过程配置项在演化过程中也可能进行分支，例如图 3 中设计阶段的过程配置项在 1．0 版本上产生了一个新的分支。这种情况下前一步骤上的演化有可能使它的多个分支随之演化。例如图 3 中分析过程配置项从 1．0 版演化到 1．1 版后，设计过程配置项的 1．0 版将受到影响，因此由 1．0 版派生出来的两个分支都必须进行演化以反映新的系统分析结果。

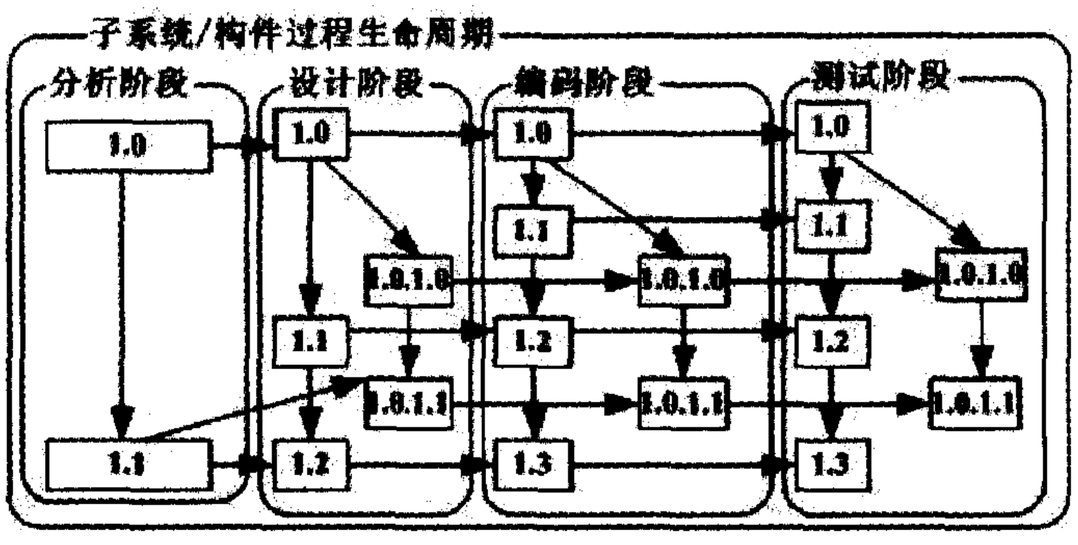

图 3 过程生命周期

3.3 过程迭代中的版本控制

过程模型的迭代执行将伴随着相关配置项的版本演化。过程步骤中进行的一系列活动将产生新的配置项或使已有的配置项发生演化。过程集成将在前一个集成版本基础上增加或删除一些过程配置项，或者更新其中一些配置项的版本。这些操作反映了本次迭代中该过程步骤的演化目标。

配置项演化过程中的每一个版本都有来源版本，过程配置项也是如此。如果某个过程步骤是本次迭代的开始，那么迭代目标就决定了本次演化的基线。此后每个过程步骤上演化的基础版本将决定于前一步骤。例如图 3 中分析过程配置项从 1．0 版演化到 1．1 版后，设计阶段进入此次迭代。此次迭代中分析阶段的演化发生在分析过程配置项 1．0 版本的基础上，与之对应的设计过程配置项版本是 1．0。因此设计过程配置项上由 1．0 版派生出来的两个分支都将进行演化，它们各自的最新版本 1．1 和 1．0．1．0 将成为此次设计阶段演化的基线。通过这样的控制策略，各步骤上的集成配置项建立起了横向和纵向两个方面的追

踪关系，使得变更影响分析、缺陷跟踪等成为可能。

4　过程模式的描述和执行

在我们的过程管理框架中，活动是版本演化的主要载体。活动只描述作什么，而过程模式为活动提供了执行方案[4]。活动的执行方案并非一成不变，因此一个活动可以采用多种过程模式。过程模式的描述和执行是最关键的问题。

4.1　配置项类型

配置管理中面临的软件资源多种多样，但对于特定的软件组织，需要处理的软件资源种类是有限的。FDSCM 引入配置项类型的概念对配置项进行结构化描述。一个配置项类型代表着一类具有相同属性的软件资源，例如图片配置项都必须描述图片格式、分辨率等。

在 FDSCM 中，活动的执行将通过配置项的演化体现。因此，配置项类型也成为过程模式的描述基础。活动以满足特定要求的配置项的创建或版本演进为目标，而过程模式为演化过程的执行提供方案。

4.2　过程模式的描述和执行

Spearmint[3]是一种形式化的过程建模工具，通过产品流的方式描述活动、实体资源以及二者之间的使用、修改和生产关系。这种建模方式强调软件资源的演化关系，适合于以产品为中心的过程描述。我们在此基础上增加了资源权限和版本控制策略，以此作为过程模式的描述手段。

过程模式描述了配置项在一系列子活动的作用下的演化关系。如图 4，活动与配置项之间存在产生、使用、修改和分支四种关系：产生表示配置项的创建；使用关系表示活动对输入的要求；修改是指该活动将对某配置项进行修改，并产生一个新的版本；分支则意味着在某个版本上产生一个新的演化分支。活动与工具之间是使用关系，表示在开发过程中使用具体的工具解决问题。配置项还可能进行集成操作，这表示多个分别开发的配置项集成后得到新的配置项版本。

过程模式通过配置项的产生、使用等关系将多个子活动连接起来为某个高层活动提供执行方案。这种描述方法仅仅规定了活动之间的隐含顺序，使得活动之间的并行执行成为可能。例如图 4 中，活动 1 产生配置项 4 后，活动 2 就可以开始，而不需要等到活动 1 完全结束。

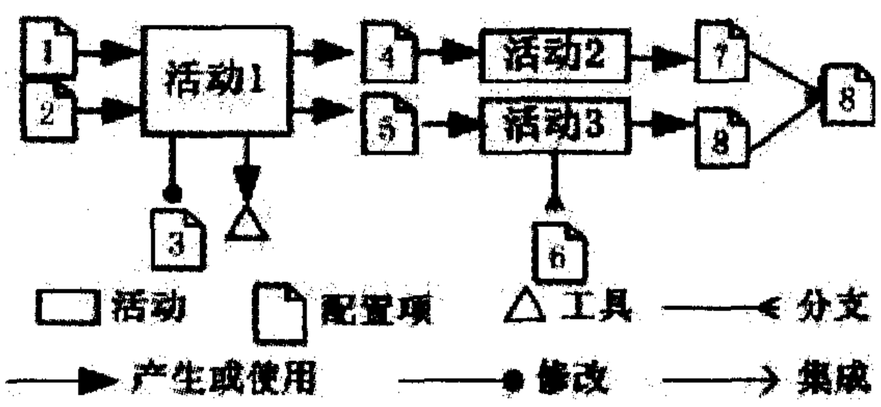

图 4　过程模式的描述

过程模式通过一系列子活动为高层活动提供执行方案。这些子活动同样可以按照某种过程模式分解为更低层次的活动。活动分解必须满足一致性规则，即子活动的执行结果必须满足上层活动的要求。例如，上层活动的修改或分支操作可能分解为子活动层上一系列产生、集成等操作。同时过程模式的外部输入不能超出该活动的输入范围，但子活动层次上配置项的创建、修改或分支的范围可能超过所对应的高层活动。

4.3　一个过程模式实例

每个软件组织在日常开发活动中都会积累起大量的过程知识，这些知识可以为本组织内相似问题的解决提供方案。我们的过程管理框架为这些知识的描述、存储和运用提供了方法。

图 5 描述了一个过程模式实例。某项目中，由于上一版本的产品性能不佳(主要是速度慢)，因此执行一个从“详细设计”过程步骤开始，以提高产品性能为目标的迭代过程。这是一个用 Delphi 开发的以数据库为中心的项目，因此在“详细设计”步骤中需要开展“数据库设计优化”活动，输入配置项为上一版本的可执行程序及相应的测试报告，在此基础上需要产生一个新的数据库设计版本。“数据库设计优化”活动可以采用过程模式“通过 SQL 分析优化数据库设计”。这个模式中使用了 Delphi 自带的数据库操作监控工具 SQL Monitor 和一种可以提供 SQL 执行分析功跑的工具 TOAD。这个模式通过截获并分析系统瓶颈部分所实际执行的数据库操作语句，从表结构设计和程序的数据库访问方案两方面提出改进意见，在此基础上与原数据库设计集成得到一个新版本(图中阴影部分)。

一个活动可以对应多个过程模式，主要取决于开发方法、开发者习惯以及可获得的开发资源状况等。例如一个加密算法的编码活动可以采用从算法描述到实现的开发模式，也可以采用从加密构件选取、适配到功能确认的复用式开发模式。

图 5 过程模式实例

5 总结

配置管理是现代软件开发过程的核心，同时也为过程管理提供了切入点。基于配置管理的过程管理通过配置项的演化规则对开发过程进行控制，一方面使得过程控制更加有效，另一方面也使得配置管理的系统性和可操作性更高。我们的过程管理框架以子系统和构件为基本的过程控制单位，在不同的层次上执行不同的过程管理策略，提高了过程管理的灵活性。

文献[11]提出将需求规格说明组织成由需求项组成的层次结构并进行版本管理，以此来控制需求变更。在我们的过程管理框架中，任何性质的变更请求都可以与配置管理统一起来。一个或多个(经合并)变更请求进入实施阶段后都将分解到某个子系统或构件上，开始一个过程生命周期。例如一个改进性能的变更请求将产生一个以"详细设计"为起点的生命周期，而一个增加功能的变更请求将引发一个以"需求分析"为起点的生命周期。它们的目标都是满足相应的变更要求，而变更请求的结果是该生命周期中产生的新的配置项或新的版本，以及由此带来的新的产品发布版本。通过这种方式，我们可以建立起从变更请求的提出到完成(配置项的演化)的完整的变更管理流程。

这个过程管理框架已经在我们的配置管理工具 FDSCM(及相应的变更管理系统)中部分地实现，其中的思想得到了较好的验证。与此同时，我们也发现了其中的一些缺陷，例如过程模式的描述和执行的易用性仍有不足，这些都将在我们的进一步工作中加以研究。

References

[1] Lindsay, P., MacDonald, A., Staples, M., Strooper, P.. A Frameworks for Subsystem-based Configuration Management. Software Engineering Conference, 2001. Proceedings. 2001 Australian, 27-28 Aug. 2001 Pages: 275-284.

[2] Susan Dart. Concepts in Configuration Management Systems. Software Engineering Institute, CMU, 1991.

[3] Ulrike Becher-Kornstaedt, Holger Neu, Gunter Hirche. Software Process Technology Transfer: Using a Formal Process Notation to Capture a Software Process in Industry. 8th European Workshop, EWSPT 2001 Witten, Germany, June 19-21, 2001 Proceedings.

[4] Michael Gnatz, Frank Marschall, Gerhard Popp, Andreas Rausch, Wolfgang Schwerin. Towards a Living Software Development Process Based on Process Patterns. 8th European Workshop, EWSPT 2001 Witten, Germany, June 19-21, 2001 Proceedings.

[5] 软件配置管理—团队开发的基石. http://www.hansky.com/cn/resources/recommended/cmm.html.

[6] 陈兆琪，钟林辉，张路，谢冰. 软件变化管理系统研究. 小型微型计算机系统，2002 Vol. 23 No. 1.

[7] 杨芙清，梅宏，李克勤. 软件复用与软件构件技术. 电子学报，1999 年 02 期：68-75.

[8] MEI Hong, ZHANG Lu, YANG Fuqing. A Component-Based Software Configuration Management Model and Its Supporting System. Journal of Computer Science and Technology, Vol. 17 No. 4, July 2002.

[9] 车向东，徐红，马云静，刘又诚. 个体软件过程实验研究. 北京航空航天大学学报，1998 年 04 期.

[10] Miro Casanova, Ragnhild Van Der Straeten, Viviane Jonckers. Supporting Evolution in Component-Based Development using Component Libraries. Proceedings of the Seventh European Conference On Software Maintenance And Reengineering (CSMR'03).

[11] Ivica Crnkovic, Peter Funk, Magnus Larsson. Processing Requirements by Software Configuration Management. 25th Euromicro Conference(EUROMICRO'99) Volume 2.

A Process Management Framework based on SCM*

PENG Xin, ZHAO Wen-yun, ZHANG Zhi

(*Computer Science and Engineering Department, Fudan University, Shanghai 200433, China*)

Abstract: SCM(Software Configuration Management) is the center of quality assurance. SCM can help to improve the effect of process control. This paper introduces the concept of process pattern and proposes a SCM-based process management framework. In the framework sub-systems and components of various granularities are the basic units of process management. It performs process control on three levels, namely process model level, team cooperation level and personal activity level. And different policies are performed on different levels. So configuration management can be more effective and process control can be more flexible.

Key words: configuration management; software process; process management; process pattern; change management

* Supported by the National High Technology Development Program of China under Grant No. 2001AA113070; Science Technology Committee of Shanghai under Grant No. 035115026

一个基于软件配置管理的过程管理框架

作者：彭鑫， 赵文耘， 张志
作者单位：复旦大学计算机科学与工程系软件工程实验室,上海,200433