

软件配置管理

1. 概述

没有任何一个行业象计算机工业发展的如此迅速。在今天的软件产业更是如此，技术和产品的更新日新月异，所有技术人员和管理人员都感到明显的压力，这种压力集中体现在两个方面：提高产品质量，缩短面市时间。现在软件产品的开发，对市场投放速度的要求成倍增长；Internet/Intranet 应用的发展改变着软件的开发，传递，和分发方式；不断提高的软件质量的要求，使越来越多的软件机构感到规范开发迫切性；多平台，多操作系统，多开发工具，多对象类型，多计算机语言等，使复杂的软件开发环境更加难以控制。

我国信息技术产业的蓬勃发展促使各种先进技术和产品在国内被广泛的应用，为国内的软件开发注入了活力。然而，值得注意的是，各种先进的操作系统，开发工具等在带来效益的同时，也使得我们的开发环境日益复杂化而难以管理。而无组织的开发环境，会导致潜在问题的产生，这些问题一般难以发现，直至出现影响整个系统的致命错误，但此时往往又为时晚矣。

软件工程使软件开发从手工作坊上升到团队开发模式，其开发工作围绕着软件生命周期的分析设计、开发、测试、运行维护四个阶段进行。通过使用软件工程的方法及工具，可以避免开发过程中许多可能出现的错误，提高软件的可重用性，降低软件测试和维护中的工作量，从而大大提高软件产品的质量，缩短开发周期。

在团队开发的模式中，软件开发管理就显得更加重要，其管理的好坏将直接影响到软件产品的质量。如果缺乏对软件开发的统一管理，势必造成以下问题的出现：

- 由于开发经费及开发时间的限制，不可能一次开发就解决所有问题，许多问题有待维护阶段解决，因此带来的是软件产品的不断升级，而维护和升级所必需的文档往往非常混乱；
- 开发商开发过程缺乏规范化的管理，即使有源程序文档也由于说明不详细而不能对产品进行进一步的功能扩充，用户不得不再投入大量的经费去开发新产品，浪费大量的人力、物力和时间；
- 在软件的团队式开发中，人员流动在所难免，如管理不善，有些人员的流动将对开发产生致命的影响。特别是软件开发管理人员或核心成员的流失，有可能造成无法确定软件产品中各模块所处的状态及阶段，使软件产品的版本出现混乱，甚至可能泄漏公司的核心机密；
- 管理不善致使没经测试的软件加入到产品中，不但影响产品的质量，有时还会导致致命的错误，造成不可挽回的损失；
- 用户与开发商没有有效的沟通手段，用户投入了开发费用后，得到的是有关可执行程序以及一堆杂乱无章的文档，即使是较好的文档，对不熟悉开发过程的专业人员来说也无从下手，更谈不上日后的维护和升级，用户的利益无法保证；

- 软件生产达不到规模化，无法生产出软件企业内部的软件标准构件仓库，使应用软件产品总处于一种低水平、重复开发的状态，不但时间得不到保证，而且成本也无法降低，使产品没有市场竞争力。

这些问题在实际开发中表现为，项目组成员沟通困难，软件重用率低下，开发人员各自为政，代码冗余度高，文档不健全等；造成的结果是：数据丢失，开发周期漫长，产品可靠性差，质量低劣，软件维护困难，用户抱怨使用不便，项目风险增加等。

2. 软件配置管理

缺乏软件开发管理，会导致种种问题的出现，这些问题使得最终开发出来的软件产品的质量难以保证，应用难以稳定。那么，怎样进行软件开发管理才能生产出高质量的软件产品呢？在 ISO9000 质量管理和质量保证标准中，制定了《在软件开发、供应和维护中的使用指南》标准，该标准除对软件生命周期的各个阶段做了严格的规定外，还在其质量体系中规定了与阶段无关的支持活动，其中软件配置管理（Software Configuration Management，简称 SCM）被放在首位。

2.1. 配置管理的概念

早在七十年代初期加利福尼亚大学的 Leon Presser 教授就撰写了一篇论文，提出控制变更和配置的概念，之后在 1975 年，他成立了一家名为 Soft Tool 的公司，开发了自己的配置管理工具：CCC，这也是最早的配置管理工具之一。之后，随着软件开发规模的逐渐增大，越来越多的公司和团队意识到了软件配置管理的重要性，而相应的软件配置管理工具也如雨后春笋一般，纷纷涌现，比较有代表性的有：Marc Rochkind 的 SCCS(Source Code Control System) 和 Walter Tichy 的 RCS(Revision Control System)，这两种工具对日后的配置管理工具的发展做出了重大的贡献，目前绝大多数广泛使用的配置管理工具基本上都是基于这两者的设计思想和体系架构。

什么是软件配置管理呢？软件配置管理有多种定义，在 1986 年出版的 Wayne Babich 的《Software Configuration Management: Coordinating for Team Productivity》一书中把软件配置管理描述为“对软件开发组所建立的软件的修改进行标识、组织和控制的艺术，其目标是减少错误，提高生产力”。这个定义比较简单，而在 1993 年出版的 Steve McConnell 的《Code Complete》一书中，从另一个角度对软件配置管理进行了定义：“配置管理能够系统地处理变更，从而使得软件系统可以随时保持其完整性。配置管理又可称为‘变更控制’，可以用来评估提出的变更请求，跟踪变更，并保存系统在不同时间的状态。”

软件配置管理是一套规范、高效的软件开发基础结构。作为管理软件开发过程有效的方法，SCM 早已被发达国家软件产业的发展和实践所证明。SCM 可以系统地管理软件系统中的多重版本；全面记载系统开发的历史过程，包括为什么修改，谁作了修改，修改了什么；管理和追踪开发过程中危害软件质量以及影响开发周期的缺陷和变化。SCM 对开发过程进行有效地

管理和控制，完整、明确地记载开发过程中的历史变更，形成规范化的文档，不仅使日后的维护和升级得到保证，而且更重要的是，这还会保护宝贵的代码资源，积累软件财富，提高软件重用率，加快投资回报。SCM 是通往 ISO9000 和 SEI CMM 标准的一块基石。

在软件开发团队中，正确地采用、实施软件配置管理系统，必将提高生产力，增强对整个项目的控制，改善软件产品的质量，从容面对快速面市和产品质量的双重压力。软件配置管理系统的实施，一般来讲要考虑两个方面的因素：流程和工具。流程和工具是相辅相成的，流程起决定性作用，它确定了管理的规则和方法，工具用来将变更存储在一个中央存储库中，可以重现任一时期的历史版本，一个好的工具可以提高效率，是贯彻实施流程的必要手段。

因此，在一个开发团队中，实施配置管理流程比采用配置管理工具更重要，我们需要充分考虑，制定出适合自己企业的配置管理流程，该流程必须与公司的开发规范、质量系统等完全结合。

2.2. 配置管理的功能

配置管理系统应该具备以下主要功能：

- 并行开发支持：因开发和维护的原因，要求能够实现开发人员同时在一个软件模块上工作，同时对同一个代码部分作不同的修改，即使是跨地域分布的开发团队也能互不干扰，协同工作，而又不失去控制
- 修订版管理：跟踪每一个变更的创造者、时间和原因，从而加快问题和缺陷的确定
- 版本控制：能够简单、明确地重现软件系统的任何一个历史版本
- 产品发布管理：管理、计划软件的变更，与软件的发布计划、预先定制好的生命周期或相关的质量过程保持一致；项目经理能够随时清晰地了解项目的状态
- 建立管理：基于软件存储库的版本控制功能，实现建立（build）过程自动化
- 过程控制：贯彻实施开发规范，包括访问权限控制、开发规则的实施等
- 变更请求管理：跟踪、管理开发过程中出现的缺陷（Defect）、功能增强请求（RFE）或任务（Task），加强沟通和协作，能够随时了解变更的状态
- 代码共享：提供良好的存储和访问机制，开发人员可以共享各自的开发资源

2.3. 配置管理的实施

实施配置管理系统，一般的步骤和需要考虑的问题如下：

2.3.1. 规划、调整网络开发环境

一个规划良好的开发环境，是实施配置管理系统的前提。在此阶段我们要对配置管理系统做出规划，主要考虑以下问题：

- 网络的带宽、拓扑结构
- 服务器的选择、命名规范
- 存储区的定位
- 开发人员及组的命名规约等

2.3.2. 设计配置管理库

根据项目开发的要求，设计开发资源的存储模式，良好的存储模式有利于减轻管理上的负担，增强配置管理库的访问性能，同时便于控制访问权限，保护软件资产。

2.3.3. 定义配置管理系统的角色

在此阶段，我们需要确定与配置管理相关的所有角色，包括他们的相应的活动。在开发过程中，一个开发人员可能兼任多种角色，但一项任务在同一时刻只能由一个角色来执行。

一般配置管理中的角色主要包括：

- 项目经理：项目经理在配置管理方面的职责是依靠配置管理员、系统管理员和系统体系结构设计人员的帮助，制定项目的组织结构和配置管理策略。这些工作包括：定制开发子系统，定制访问控制，制定常用策略，制定集成里程碑，以及进行系统集成
- 配置管理员：配置管理员的职责是根据项目经理制定的开发组织结构和策略，实施、维护配置管理的环境。其主要职责如下：创建配置管理库，对存储库进行日常备份和恢复，维护配置管理环境，及管理配置管理相关的用户
- 软件开发人员：软件开发人员依据项目的开发和配置管理策略，创建、修改和测试开发工件
- 集成人员：对软件进行归并，形成相应的基线或发布版本
- QA 人员：需要对软件配置管理有较深的认识，其主要工作是跟踪当前项目的状态，测试，报告错误，并验证其修复结果

2.3.4. 制定配置管理流程

这是配置管理实施的一个重要阶段，其主要目的是根据项目开发的需要，制定相应的配置管理流程，以更好地支持开发，主要活动包括：

- 定制并行开发策略：合理的并行开发策略应该具有以下特点：协调项目的复杂性和需求，统一创建分支类型和元数据，为开发过程中的变更集成制定有效的规范，适时反映开发过程中方法和需求的变化
- 发布版本管理：软件开发过程中的一个关键活动是提取工件的相关版本，以形成软件系统的阶段版本或发布版本，我们一般将其称为稳定基线。一个稳定基线代表新开发活动的开始，而一系列定制良好的活动之后又会产生一个新的稳定基线。有效地利用此项功能，在项目开发过程中可以至始至终管理、跟踪工件版本间的关联。

2.3.5. 相关人员的培训

一般来讲，实施配置管理系统，相关人员需要接受一下培训：

- 管理员培训：针对配置管理员，主要学习配置管理工具管理相关内容
- 开发人员培训：针对开发人员，主要学习配置管理工具与开发相关的常用操作
- 管理流程培训：针对全体人员，目的是了解配置管理策略和流程，以及如何与开发管理、项目管理相结合

以上只是简单地介绍了配置管理系统实施的相关内容，软件配置管理作为软件开发过程的必要环节和软件开发管理的基础，支持和控制着整个软件生命周期。若要有效的实施软件配置管理，首先要通过一系列的培训，培养软件开发者的管理参与意识，同时更重要的是借助已有的经验教训，建立起真正适合自己团队的管理流程。

3. CMM（软件能力成熟度模型）与 SCM（软件配置管理）

在 COCOMO 模型中，对一个项目团队的软件开发能力作了如下定义：

软件项目的开发能力=（团队技能）（工具）（规模）（过程）

其中：

- 团队技能：开发团队的技能、经验，以及精神
- 工具：开发团队使用各种工具的自动化程度
- 规模：开发团队的人员规模，以及人为因素造成的复杂程度
- 过程：团队软件开发的方法、标记及成熟度

基于此模型，我们可以看出要提高团队的开发能力，可以从以下四个方面入手：

- 降低问题的复杂性，减少开发团队规模：尽可能多采用第三方现成的组件，多采用软件复用规范标记法，基于模型的工件，或由工具直接产生工件
- 改进过程的效率和成熟度：吸纳新技术、新工具、新经验
- 提高团队技能的技术熟练程度：培训、指导、顾问咨询
- 扩展工具的自动化程度：双向工程分析、评估自动化

在这四个方面中，软件开发过程的改进最为重要，目前国际上有多种改善或评估软件开发过程的方法和工具，其中以卡耐基梅隆大学的软件工程研究所（SEI）提出的软件能力成熟度模型（Capability Maturity Model, CMM）最具影响力。CMM 将软件过程成熟度划分为 5 个级别，从一级到五级分别为：混乱级、可重复级、定义级、管理级和优化级。CMM 级别越高，就表示软件开发过程越规范。每个成熟级别由一系列关键流程领域（Key Process Areas, KPA）组成，每个 KPA 确定一组相关活动。当这些相关活动一起开展时，它们完成一系列被认为在该成熟级别建立流程能力有重要影响的目标。

在 CMM 二级中，有以下六个 KPA：需求管理、软件项目规划、软件项目跟踪与勘察、软件分包管理、软件质量保证、软件配置管理。其中最后一个 KPA 软件配置管理的目的就是在项目的整个软件生命周期内建立并维护软件项目产品的完整性，更好地管理开发。实际上，软件配置管理是大多数软件工程和管理流程的一个重要构成部分。

针对配置管理，在 CMM 中仅仅定义了一系列的流程，而没有对实施方法提出任何要求。因此，如果要达到 CMM2 的要求，我们需要自己建立配置管理流程，并选用相关的工具，来贯彻流程的实施。

4. 总结

实施配置管理方法及工具是一项系统工程，同时也是一项领导工程，因为任何新的工具和方法的引进，必将在初期导致团队生产力的短期下降，在此过程中，需要团队的领导的魄力和决心，以及团队成员的理解和配合，大家共同努力，同时可以借助一些外来的帮助，例如专家的顾问咨询等，以尽可能减小负面影响的程度，缩短负面影响的时间。在度过这段下降区后，就能体验到配置管理系统所带来的一系列好处。