

从混乱到有序—版本管理与青鸟配置管理系统概览

李 欣

如果你是一个专业的软件开发者，你一定曾经为这样的问题苦恼过：昨天刚改过的一个源程序，今天怎么也找不到了，拿到的都是一些老版本；刚改完一个源程序，却发现你要改的 bug 开发小组的其他组员已经在另一个版本中改过了，或者是他已经把这个文件改的面目全非，你根本不可能把两者合二为一；还有当你想发布你的软件时，却怎么也判断不出哪些文件版本能够组成这个发布版。这些问题都是所谓的版本管理问题。虽说这些问题通过严格管理可以解决，比如为每个源文件建个目录，并且把该文件的新老版本都按时间顺序编上号，不允许多个程序员同时修改一个文件等等。但一个项目组就几个人，要手工管理数以百计的文件版本既费时又费力，难免会出现各种纰漏，而且这样也难以有效组织项目组的协同开发。

一、版本管理与版本管理工具

为了便于对版本管理问题及其解决方法做深入的讨论，也为了使讨论更加规范化，我们先来定义一些主要概念。

版本管理是针对软件开发过程中涉及到的各种软件资源进行的管理。而软件配置管理（SCM）是系统开发过程中的是一种按规则实施的管理软件开发和维护过程以及软件产品的较完整的方法。版本管理是配置管理的核心。软件配置管理系统在软件质量管理中起着重要作用，是 CMM 与 ISO9000 质量管理体系的核心部分之一。因此，能否实现有效的版本管理，现在已经成为判断软件企业是否专业化和正规化的重要标准。

版本管理涉及到任务主要包括，保存软件开发过程中待开发系统的各种中间状态，控制对软件系统的修改，记录软件系统的变化过程。版本管理工具是指能对版本管理提供自动化支持的软件工具。一般而言，版本管理系统分为版本库和临时工作区两部分。版本库负责统一存储各种软件资源及其各种版本，临时工作区则是软件开发者修改（编辑、编译、调试等）软件资源的工作场所，版本管理系统则是完成软件资源在版本库和临时工作区之间的存取工作。版本管理系统利用统一的命名规则标识软件资源诸多版本，避免了版本紊乱现象。版本管理系统对存取软件资源采取加锁等控制策略，保证了多开发者共同开发时软件资源的内容一致性和正确性。要利用版本管理工具辅助软件开发，必须按照版本管理的规则进行工作。相应的软件开发模式是：

“检出（Checkout） → 修改（Edit） → 检入（Checkin）”

其中：检出是版本库中取出计划要修改的软件资源，将其放入临时工作区中；随后用户可以在临时工作区中进行修改工作；当用户工作完成时，通过检入操作将工作区内容保存到版本库。在版本管理系统的控制下，修改产生的新版本将存储在版本库中，并保持、维护与原系统的关系（版本关系，参照关系等）。

虽然在国外利用版本管理工具辅助软件开发已经成为一种基本要求，但由于国内的大部分软件企业直到最近才开始认识到版本管理的重要性，而且市场上能够对版本管理提供自动化支持的成熟工具并不多，因此在项目开发中引入版本管理目前还只是一种发展趋势。但我们相信，在不久的将来，甚至就在三五年内，在国内软件企业自身不断发展和与国际接轨的要求下，这一趋势必将成为一种潮流。本文通过介绍使用青鸟配置管理工具实现版本自动化管理的原理与方法，希望使大家对版本管理的重要性有一个初步的了解。

青鸟配置管理系统（以下简称 JBCM）是北京大学软件工程研究所在青鸟攻关工程中研制的基于构件的新一代软件配置管理工具。作为青鸟软件生产线的重要组成部分，青鸟配置管理系统面向软件企业开发、管理相关资源的需求，支持基于构件的软件开发方法，符合 ISO9000 和 CMM 软件质量保证体系的要求，可以有效地改善软件企业的开发过程。JBCM 按照系统的不同面向（项目组工作需求，企业开发管理需求）分为两个不同级别的系统。

项目组级系统——JBCM/Team(JBCM/T)。JBCM/T 是面向项目小组级别开发、管理需求的系统，主要包括基于构件的版本管理、团队协同开发支持功能、配置管理，以及人员、权限管理和基本的审计、统计功能等。

企业级系统——JBCM/Enterprise(JBCM/E)。JBCM/E 是面向软件企业级别开发、管理需求的配置管理系统。JBCM/E 可以全面支持企业实现 CMM 与 ISO9000 质量管理，可以完整地存储、管理企业内部资源及其重要关系，改善软件企业的开发过程。

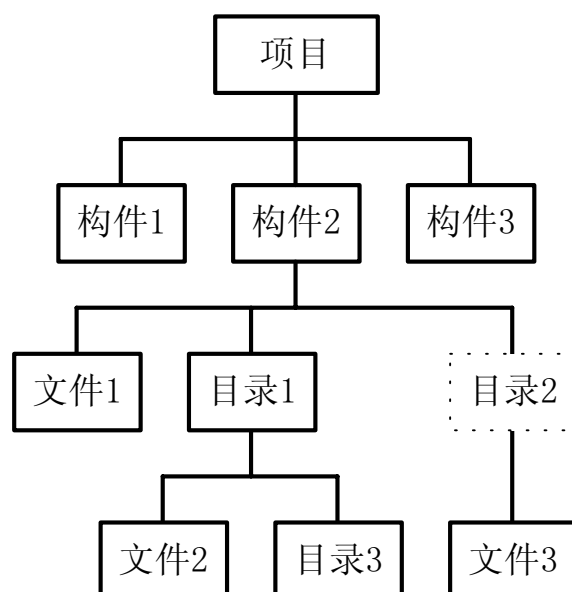
JBCM/T 的学习版现在可以免费下载，网址是 <http://se-lab.cs.pku.edu.cn/95/jbcm>。下面就以该学习版为例来介绍 JBCM/T 的原理和使用方法。JBCM/E 因为概念较复杂，且非常专业化，所以在本文中就不介绍了。

二、青鸟配置管理系统原理

1. JBCM 的软件开发模型——项目、构件结构

在 JBCM 系统中，设想软件开发主要分为两个层次：项目和构件。项目对应于一个独立的待开发的软件系统，构件则对应于软件系统中的一个比较独立的开发部分（比如说子系统）。一个软件项目由相关构件组合而成。在 JBCM 中推荐使用项目/构件结构来进行软件开发。在某些设计中，构件自身还可能包含其他构件。JBCM/T 所支持的项目/构件结构是两层结构，即项目可以包含一个或多个构件，而构件不可以包含其他构件，也就是说项目之下的所有构件都是平级的。

一个项目可以有自己的内部结构，它的内部结构就是各个构件。一个构件也可以有自己的内部结构，在 JBCM 中我们把构件之下的结构称为目录。即一个构件中可以包含若干文件也可以包含若干目录，目录之下也可以包含若干目录或文件。JBCM 系统构件中的目录对应于文件系统中的目录组织结构。构件的具体结构如下图所示。



一般来说，构件应该是项目中的一个在逻辑上相对独立的开发单位，但在实际应用中可以有所变通。同时，注意到构件是 JBCM 进行版本管理的基本单位，在划分构件时将版本变化相对同步且一致的部分放在一个构件中会有益于系统的管理和开发过程中的使用。常见的构件划分方法可有下列三种。

(1) 一个构件对应项目中的一个模块——通常，一个项目总是由若干逻辑上相对独立的功能模块组成，在开发时就可以根据分析和设计文档直接为每个模块创建一个构件，由一个构件开发组中的成员共同开发。

(2) 一个构件对应一个可执行文件——某些软件项目具有一种工具包的特征，即在这个软件项目中包含若干独立的软件工具，这些软件工具既可以独立执行，也可以协同工作，通常在项目中还会提供一个集成环境把这些工具集成起来。如 VC 和 Norton Utility 就是这样的软件。在开发这样的软件项目时，可以让一个构件对应项目中的一个软件工具，即一个可执行文件。

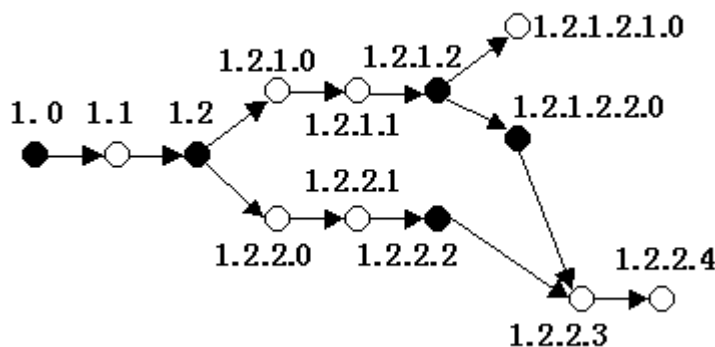
(3) 一个构件对应若干模块公用的部分——在模块式开发中，可能在若干模块（构件）中存在一些公用的部分，如公用的代码或分析设计文档。这些公用的部分很可能并不是一个逻辑上独立的单位，但是这些公用部分的变化必须立即反映到每个使用它的构件中，如分析设计文档的变化要求所有涉及的构件都立即作出相应的变化。这时，可以把这部分公用的东西放在一个单独的构件中，单独维护，其他构件则引用它的内容。

2. 版本管理

JBCM 的版本管理主要分为两个层次：文件的版本和构件的版本。

修改是软件开发及维护中最基本的活动之一。引起修改的原因是多种多样的，例如不断发生变化的需求，对错误的修正等。但是开发和维护人员对软件的修改并不是一蹴而就的，这种修改会不断的发生新的错误，从而具有相当的反复性。为了保护开发维护人员的劳动成果，JBCM 把开发过程的中间状态作为版本保存下来，以便日后随时提取。为了保证系统对版本的管理，JBCM 规定：开发人员要进行修改时，必须首先“Check out”相应的构件；修改完毕后，必须把构件“Check in”才能使这些修改反映在构件和项目中。每一个“Check out/Check in”过程都产生构件的一个版本，同时也可能产生若干文件的版本。使用者可以在恰当时刻对构件的某个版本进行分支，建立一个新的开发流，以此适应于不同的开发需求。例如开发适用于不同操作系统的同一应用软件。构件多个分支还可以进行合并。由此，一个构件的所有版本构成了一棵版本树。

文件的版本结构主要记录了文件的变化过程，它以版本树的结构跟踪记录文件的变化。在版本树中，每个节点都是直接从其直接父节点直接修改而来的。一个父节点可以拥有一个或多个子节点，每个子节点反映文件的一个变化方向。文件的版本结构对开发者来说是不可见的，因为 JBCM 版本管理的基本单位是构件，因此开发者所能看见的版本结构是下面要谈到的构件版本结构。



构件的版本结构与文件的版本结构相似，都是由版本树来表示，如上图。在构件中，引入了分支的概念，每个分支都是由其直接父节点而来，每个分支有自己的版本变化树，反映了构件的一个变化方向。构件的版本记录了构件的发展变化过程，而构件是由文件组成的，因此构件的版本就是由构件中所有文件的版本组合而成。构件组成文件的增删以及其中任何一个文件的修改都被视为对整个构件的修改。

3. 用户控制

为了更好的支持多人并行开发，JBCM 采用自己的用户管理策略。JBCM 中的用户分为三个层次：系统用户、项目用户和构件用户。系统用户是所有使用 JBCM 系统的人员，他们可能参与项目的开发，也可能只参与项目的管理。项目用户是系统用户中参加某个项目开发的人员。构件用户是从项目用户中选择出来的参与某个构件开发的用户。构件用户必须首先是项目用户。构件开发小组的成员就是相应构件的构件用户。

在实际系统中用户在不同层次具有下列不同身份：

系统管理员——系统管理员的主要任务就是管理系统用户。具体说来，系统管理员可以增加、删除系统用户，可以更改任何系统用户的口令和权限（如把一个普通用户改为项目管理员级用户）。

项目管理员和项目管理员级系统用户——在系统用户中，有一类用户拥有特殊的权限，他们就是项目管理员级系统用户。项目管理员级系统用户可以创建和删除项目（只能删除自己管理的构件）。一旦创建出一个项目，创建项目的用户即自动成为项目的管理员。项目管理员可以增加和删除项目中的用户（不能删除自己），还可以改变项目用户的权限。

构件管理员和构件管理员级项目用户——项目管理员从系统用户中选择一部分作为项目用户，项目管理员可以在这部分用户中挑选一些作为构件管理员级用户。构件管理员级用户可以在项目中创建和删除构件（只能删除自己管理的构件）。一旦成功创建一个构件，创建构件的用户即自动成为构件的管理员。构件管理员可以增加和删除构件中的用户（不能删除自己）。

系统、项目和构件中的普通用户——除了上面提到的三类用户之外，系统中剩余的用户就是普通用户了。系统的普通用户只有在成为了项目用户之后才可以参与项目的开发，同理项目用户也只有在成为了构件用户之后才可以参与构件的开发。

4. 权限和操作模式

即使都是构件的开发者，他们也有不同的分工。JBCM 系统可以设置用户对构件中文件的存/取/修改权限，并通过相应机制支持团队的并行开发活动。操作权限规定了用户可以对文件的访问权限，包括只读和可写两种权限。通过详细规定用户对文件能够对文件的访问权限，就可以控制用户的访问范围，防止用户越权访问造成混乱，从而保证软件开发的顺利进行。

用户在 Check out 构件时，可以在其权限范围内根据需要采用不同的操作模式对每个文件进行实际操作。操作模式有三种：

只读——用户只是从配置库中读取指定文件的内容，而不准备对文件加以修改。只读的文件不必 Check in。

排他写——用户打算对文件加以修改，而且规定自己在修改时，其他用户不能修改此文件。排他写的文件应该 Check in，但用户如果对此修改不满意，可以用取消 Check out 操作。

共享写——用户打算对文件加以修改，但允许其他用户也同时修改此文件。共享写的文件可以 Check in，也可以取消 Check out。Check in 时，如果另一个修改此文件的用户先于自己 Check in 了，则系统会将此文件的这两个版本自动合并。

通过权限与操作模式的设置，在加上版本管理系统对不同文件操作模式采用不同的加锁机制，就可以有效地进行项目的多用户协同开发。JBCM 建议用户谨慎使用共享写操作模式，而应该尽量使用只读和排他写这两种操作模式，这样可以最大限度地减少并行开发中的冲突现象，从而减少不必要的协调工作。

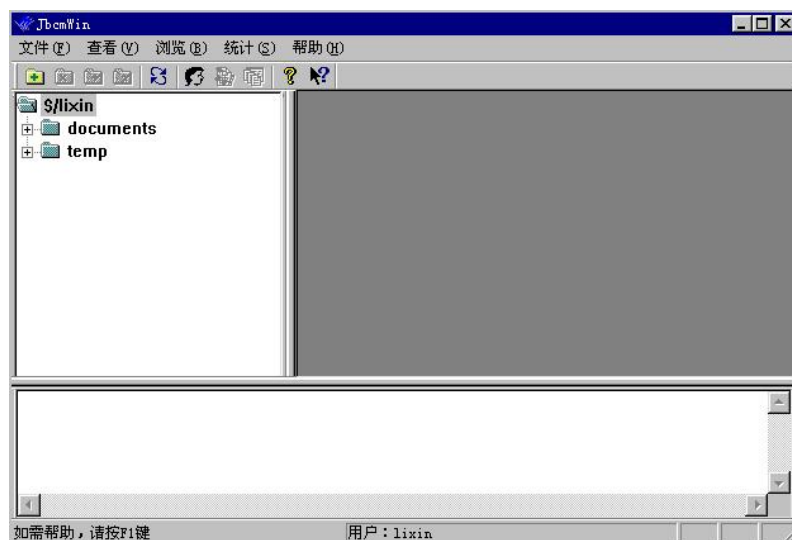
三、青鸟配置管理系统使用指南

1. JBCM 系统结构及环境需求

JBCM 系统支持 Windows NT 和 Win95/98 开发平台。系统采用客户机/服务器模式，分为服务器程序和客户机程序两部分。服务器程序集中管理、存储相关的软件资源，控制并支持团队开发工作。它通过 JBCM 客户机程序向用户提供服务，管理开发过程中的永久存储区。客户机程序可以通过 TCP/IP 协议访问服务器端软件资源，维护用户工作区与永久存储区的一致性。对于各种开发环境，如 MS Visual studio、Inprise Delphi、PowerBuilder，以及 Symantec 等，JBCM 作为一种环境服务器，通过客户机的工作区支持系统的编译、调试等开发过程，而在服务器端管理、维护系统的开发、演化历史，以及并行开发的一致性控制。

系统初始安装过程中可以设置服务器的配置库的根目录、系统管理员名称、口令等。

JBCM 客户机的初始界面为：



2. 基于 JBCM 的软件开发过程

在成功安装 JBCM 系统后（包括服务器和客户机两部分），用户可以基于 JBCM 系统进行软件开发。

JBCM 系统的基本使用过程是：登录→工作→断开。

- 登录：使用系统之前，首先要利用客户机系统登录到版本管理服务器上。首先启动服务器并确保服务器已经进入正常工作，随后可以启动客户机系统。用户选择菜单中的“文件/登录”，激活下面的对话框。在对话框中相应的位置输入服务器名称（也可以是服务器的 IP 地址）、用户名和登录口令，再单击确定即可。
- 工作：管理员可进行人员管理，权限设置等工作；开发人员则可以通过客户机浏览相关项目/构件，及其演化历史等信息，选定构件并对其开发。对构件的开发过程符合一般的基于配置管理系统的软件开发过程。一般的开发人员通常工作模式是“Check out”→进行工作→保存工作结果→“Check in”。
- 断开：使用完毕后，一般应该断开与服务器的连接，使服务器可以释放相应的资

源。退出时，选择菜单中的“文件/断开”即可断开与服务器的连接。因为其他用户可能还需要使用，所以服务器程序不必关闭，

3. 项目结构视图

登录服务器以后，在主窗口左边有一个表示项目结构的树图（见客户机初始界面）。在执行有关项目、构件的各种操作时（如增加、删除用户和构件），需要首先从此视图选择一个项目或构件，再从菜单中选取相应的操作命令。

4. 构件版本视图

构件的版本视图显示了构件的演化过程，要察看一个构件的版本视图，只需在项目结构视图中选中一个构件，再选择“浏览/版本”菜单项即可。

5. 构件结构视图

构件结构视图显示选定构件版本中的具体内容：包含的文件和目录。要打开此视图，在构件版本视图中选择一个构件的版本，再选择“浏览/结构”菜单即可。

6. 创建项目和构件

项目管理员级系统用户可以创建项目，项目管理员和构件管理员级项目用户可以创建构件。用“文件/创建项目”和“文件/创建构件”菜单可以创建项目和构件（使用相应的右键功能可以快速完成）。

7. 设置构件工作目录

在成功创建新构件后，就可以对构件进行操作了。在执行各种版本操作之前，首先要设置构件的工作目录。构件的工作目录即是构件对应的在用户开发环境中的工作区，由用户硬盘上的相应路径名指定。选择“版本管理/设置工作目录”菜单项即可执行该功能。

一个构件在不同用户环境可以位于不同的位置。因此，每个有权的用户对同一构件都应设置其特定的工作目录。

8. Check out、Check in

选定构件，并且该构件已设置了工作目录之后，就可以对构件进行 Check out 和 Check in 操作了。

Check out 时，只能选择某一支上的最新版本进行 Check out。选择“版本管理/Check out”菜单项执行此功能。下图的 Check out 对话框中，显示了构件的名称、Check out 的构件版本和目标目录。目标目录就是构件的工作目录，构件当前版本中的所有文件都将 Check out 到工作目录中。Check out 时可以通过“用户自定义”选项修改对构件中各个文件的 Check out 模式。



对构件的修改完成后，就应该把文件放回配置库中。Check in 完成的就是这个任务。选择“版本管理/Check in”执行此功能。

如下图，Check in 时可以向构件中添加文件（单击“添加新文件”）；或从构件中删除某些文件（单击“删除文件”）；也可以通过选择“取消 Check out”，用户可以指定一部分文件不 Check in，对这部分文件的修改不会反映在配置库里，JBCM 系统会直接取消对这部分文件上的锁。保持 Check out 检查框选中后，构件中的文件在 Check in 后，仍然保持 Check out 状态，用户可以继续对其进行修改，向构件中增加文件：



通过上面的步骤设置完毕后，单击详细资料用户可以观看本次 Checkin 的具体细节，确认后，回到 Checkin 主对话框，单击确定即向服务器提交当次 Checkin。

9. 用户列表

进入系统后，用户用“浏览/用户”菜单可以观看系统中的用户列表。如果在项目结构视图中选中的是表示配置库的根，则显示系统用户列表；如果选中的是一个项目，则显示项目用户列表；如果选中的是一个构件，则显示构件用户列表。

10. 增加和删除用户以及修改用户权限

系统管理员、项目管理员、构件管理员分别可以增加和删除系统用户、项目用户、构件用户。项目管理员在增加项目用户时，只能从当前的系统用户中选择；同样，构件用户在增加用户时，只能从当前的项目用户中选择。

系统管理员可以赋予系统用户两类权限：项目管理员级和系统普通用户级。项目管理员可以赋予项目用户两类权限：构件管理员级和项目普通用户级。构件管理员不可以修改构件用户的权限，构件中的用户除了构件管理员以外，都是普通用户。

增加用户的菜单命令为“用户/添加用户”，修改用户权限的菜单命令是“用户/修改权限”。

11. 修改文件操作权限

构件管理员可以根据开发的要求随时修改用户的文件操作权限。用命令“编辑/修改操作权限...”调出下面的对话框执行功能。



12. 修改文件缺省操作方式

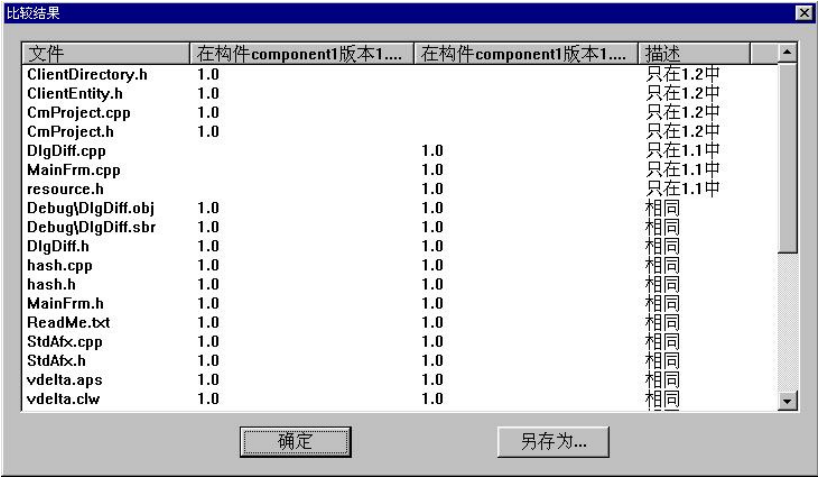
用户可以根据开发的要求随时修改自己对文件缺省操作方式。用命令“编辑/修改缺省操作方式...”调出修改文件缺省操作方式对话框执行功能。

13. 浏览构件信息和版本信息

JBCM 系统提供关于项目、构件等的审计、统计信息。选定一个项目后，通过启动菜单项“统计/项目信息”，可以查看项目创建的时间、创建者，以及用户注明的一些信息。同理，选定构件的一个版本后，也可以通过启动菜单项“统计/构件信息”和“统计/版本信息”查看相应的系统信息。

14. 构件版本比较与文件内容比较功能

在 JBCM 系统中，提供了构件版本和文件内容的比较功能。
构件版本比较功能作用于同一构件的两个不同版本间。选定一个构件，启动菜单项“工具/构件版本比较”，可以在弹出窗口中选择构件的不同版本，确定后系统将给出构件版本内所有文件的版本情况，以及其中的相同/相异情况。下图为构件版本比较的结果。



文件	在构件component1版本1....	在构件component1版本1....	描述
ClientDirectory.h	1.0		只在1.2中
ClientEntity.h	1.0		只在1.2中
CmProject.cpp	1.0		只在1.2中
CmProject.h	1.0		只在1.2中
DlgDiff.cpp		1.0	只在1.1中
MainFrm.cpp		1.0	只在1.1中
resource.h		1.0	只在1.1中
DebugDlgDiff.obj	1.0	1.0	相同
DebugDlgDiff.sbr	1.0	1.0	相同
DlgDiff.h	1.0	1.0	相同
hash.cpp	1.0	1.0	相同
hash.h	1.0	1.0	相同
MainFrm.h	1.0	1.0	相同
ReadMe.txt	1.0	1.0	相同
StdAfx.cpp	1.0	1.0	相同
StdAfx.h	1.0	1.0	相同
vdelta.aps	1.0	1.0	相同
vdelta.clw	1.0	1.0	相同

文件比较功能可以针对两个文本文件比较其差异。选定一个构件，启动菜单项“工具/文件版本比较”，可以在弹出窗口中选择待比较的文件。文件可来自用户本地存储区（如用户硬盘），或来自 JBCM 系统的配置库中，在选定比较的文件，执行确定操作后，系统将以一个明确的两文件对照图方式显示文件差异。注意：文件比较功能只能针对文本文件，比较是以行为单位进行的。