

## Firefly – 面向项目的新一代配置管理工具

配置管理工具是每一个软件开发团队必备的工具之一。正确地使用好配置管理工具可以在很大程度上提高团队的工作效率，促进成员之间的沟通；避免因版本混乱而为客户和公司造成的损失。汉星天公司（Hansky Ltd.）最新的配置管理工具 Firefly 是在总结了过去 20 多年中配置管理工具发展的趋势与技术之后，开发出的一套全新的面向项目的配置管理工具。相对于传统的版本控制工具（CVS、VSS、PVCS、ClearCase 等），Firefly 具有无可比拟的优势，下面就对这一问题进行简单的阐述。

### 1. 早期基于文件的版本控制工具的不足之处

早期的配置管理工具（例如 RCS、SCCS，包括其后的 CVS）主要是以文件为单位进行一系列的版本控制操作。所有的文件均存储在一个中央的存贮库（Repository）中，一般用户的工作模式是：**checkout->modify->checkin**。使用以文件为基础的配置管理工具可以满足一般的文件级的版本控制功能，但是不能很好的满足高级的功能需求，并造成宏观管理上的混乱。为了快速适应市场的变化和不同客户的需求，软件开发团队需要越来越多的使用并行开发模式。同时，并且为了确保软件质量，公司需要在整个项目的级别上来有效的跟踪变化的发生和对变更进行控制。针对并行开发的问题，一般版本控制系统的做法是让用户直接在**文件**上作出一个分支（Branch），使得不同用户可以同时在文件的分支和主干上对文件进行修改，而不互相干扰，有效的起到隔离开发的作用。如图 1 所示：

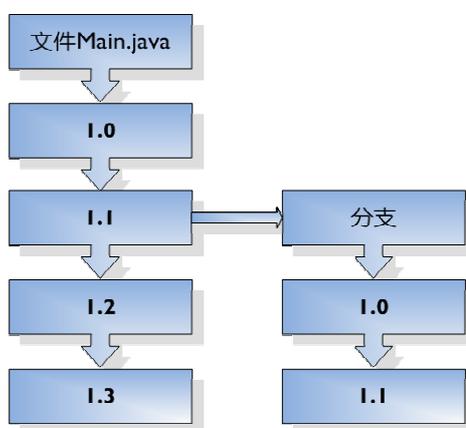


图 1：基于文件的分支

从表面上看起来这样很合理并且操作简单，但是基于文件的分支会导致以下问题：

1. 当分支生成的时候，有可能去修改每一个文件，在其中加入分支信息（如 CVS），如果文件数目较多，该操作会严重的影响系统的执行速度；
2. 每个程序员都可以对文件生成新的分支，会给项目管理带来很大的麻烦：项目经理不知道每一个分支的用途，不知道在哪两个分支之间应该作归并（Merge）；
3. 存在重复归并（Repeat Merge）的问题；
4. 文件分支之间的归并依赖于每一个程序员，不仅不可靠，还增加了程序员的工作负担。

基于文件的版本控制系统可以准确的记录每一个文件的历史，但是不能记录整个项目的历史。这会为项目管理带来很大的不便。例如，项目经理不方便知道“昨天有哪些

程序员对哪些源程序进行了修改？”（当然，也可以遍历每一个文件，查询需要的结果，但是执行速度会相当的慢）。实际的软件开发项目中，作为一个项目的整体需要具有一条清晰的代码线（Code Line），需要能看到该代码线的逐渐演进过程，而不仅仅是文件的历史记录。

## 2. Firefly — 面向软件开发项目的配置管理工具

汉星天（Hansky Ltd.）公司的最新版本控制工具 Firefly 是新一代的配置管理工具，它完全弥补了以往基于文件的版本控制工具的不足，成为软件项目开发的有力助手。

首先，在 Firefly 中，“分支（Branch）”具有了更加合理的意义，它不再是一个文件一级的概念，而是扩展到了项目级别。我们不说“某某文件具有某某分支”，而是说“某某分支具有某某文件”。从静态的角度来讲，Firefly 中的分支可以具有自己的层次结构和父子关系，分支中包含受控文件和目录；从动态的角度来讲，每一个分支都具有自己的历史，可以作为一个软件开发项目的代码线（Code Line）来进行维护。我们称 Firefly 中的分支技术为“FleXtream”。

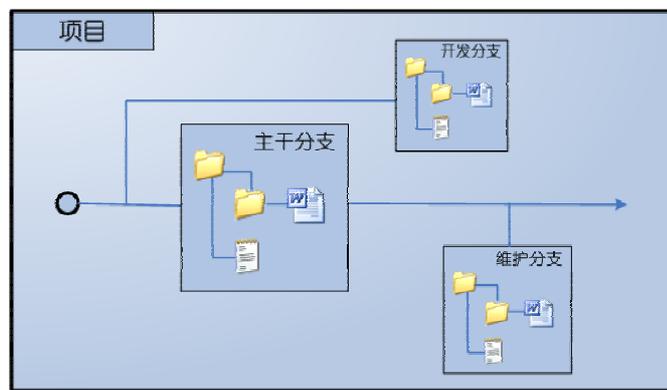


图 2: Firefly 中的分支示意

Firefly 是真正意义上的“面向软件开发项目的配置管理工具”。用户可以在项目级别清楚的查看该项目具有的所有分支（包括私有工作空间：中继工作区 Staging Workspace），以及每一个分支的目的。用户可以为分支取一个具有明确意义的名字，而不是系统生成的无意义的数字。

当一个程序员参加到项目的开发工作中时，项目经理只需要告诉他应该工作于哪个分支即可。程序员在初始化本地工作区的时候选择该分支，之后所有的 Bringover 和 Putback 操作（类似 Get Latest Version 和 Check in）都是针对于该分支的。通过对分支的 Bringover 和 Putback 操作，可以完成对文件的版本控制并达到在团队间协同工作的目的。程序员可以清楚地知道目前自己是工作在哪一个分支下（以及该分支的作用），而不是必须知道每一个文件是工作在哪一个分支下。同时，程序员还可以清楚地得知其它工作在该分支下的团队成员。分支的维护是由项目经理来做的，普通的程序员只需知道分支的名字，不需要了解其它更多的和配置管理相关的工作细节。这就避免了由程序员随意建立分支而带来的混乱。

Firefly 的 FleXtream 技术可以很好的处理并行开发的问题。当遇到需要分支的情况时，项目经理可以方便的基于旧有的分支或标记（Label）来新建立一个分支。在这两个分支上所从事的开发工作完全是独立的，这是两个互不干扰的开发代码线（一般来说，主分支是开发分支，新建的子分支为 bug-fix 分支）。项目经理还可以使用 Firefly 的

Server Explorer 工具方便的对两个分支进行内容比较和归并。Firefly 可以自动判断出文件的更新、移动、删除、重命名等操作，并且可以准确的判断出文件的版本冲突。分支的归并以项目经理为主，选择需要归并的内容，非常方便进行有效的归并管理；并由程序员配合解决在分支归并中出现的文件冲突，在归并之后，进行构建（Build）和测试（Test）。

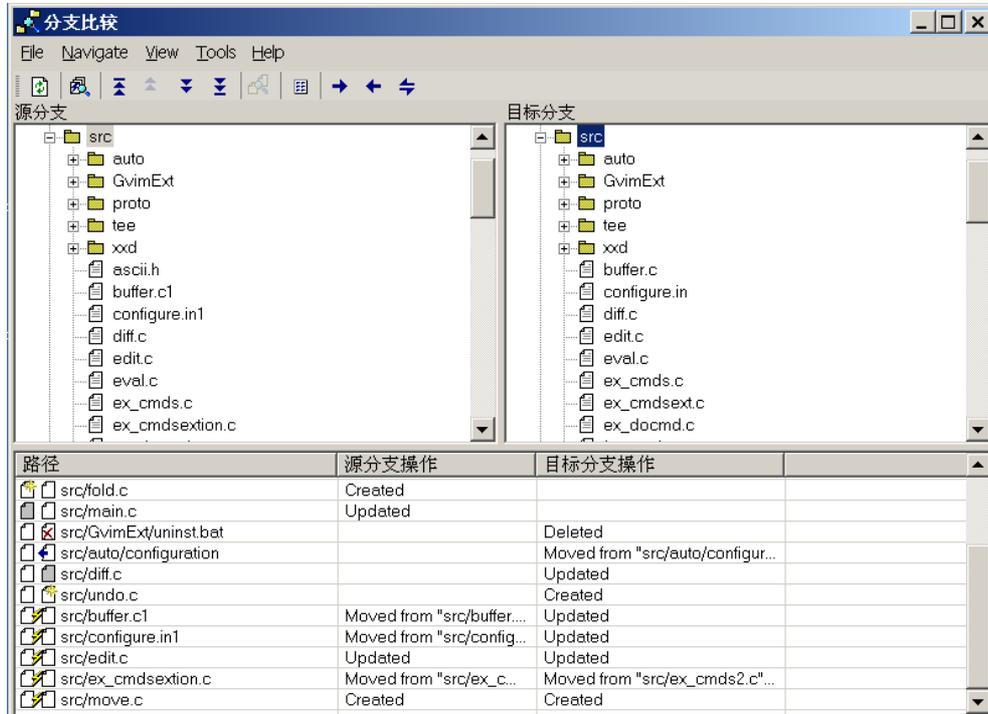


图 3: Firefly 中的分支比较

在 Firefly 中，新分支的创建非常迅速。这是因为在分支创建过程中，并不对分支中的文件进行拷贝，而仅仅是对原有分支（或标记）的状态进行了简单的快照。这不仅保证了命令的执行速度，还大大节省了硬盘空间。

在 Firefly 中，分支的历史是由若干的“变更集 (Changeset)”组成的。每一个变更集包含为了某一目的（任务）而作出的所有文件改动。一个变更集可以关联到变更管理系统 (Change Request Management System) 中的一个 Bug、RFE 或其它变更请求。由于变更集的使用，用户可以从整个项目的角度进行配置管理和跟踪变化，而不是仅仅可以查看单个文件的历史。

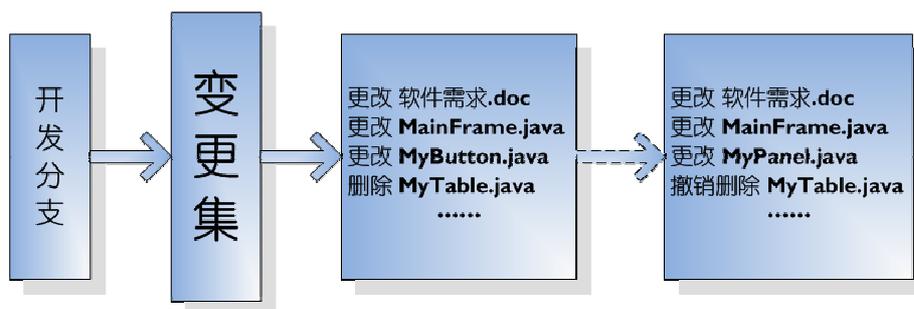


图 4: Firefly 变更集示意图

在图 4 的变更集中，包含了一组相关的文件版本改变，用户可以清楚的查看变更集中每一个文件的改变。从技术的角度来讲，变更集是实现“原子操作(Atomic Operation)”的有效方法。一个变更集中的所有变化要么全部提交，要么全部失败，不存在任何中间状态（用户提交 10 个文件，其中五个提交成功，另外五个失败）。这对于保持数据的完整性和一致性具有重要的意义。

使用 Firefly 的 FleXstream 技术，用户可以方便的实现各种 SCM 分支模式：按版本分支；按目的分支；为了修改 Bug 而分支等等。用户可以方便地建立分支，在本地工作区中修改文件、测试，之后由项目经理（有程序员配合）进行分支之间的归并。

“标记(Label)”是配置管理中一个非常重要的概念，主要是为了记录项目开发状态和回溯文件历史。在基于文件的版本控制系统中，标记是打在文件上的，如果用户在创建标记之后，删除了某个被标记的文件，则在日后不能重现标记的状态。在 Firefly 中，标记的管理也是项目一级的。标记可以完整、正确地记录项目的历史状态，不会发生任何丢失现象。

除了能够很好的支持并行开发以外，Firefly 还可以很好的支持并发开发(Concurrent Development)。即两个不同用户对同一个文件的同一版本同时执行了 Checkout 操作，这样，当第二个用户 Check in 该文件时，就会发生冲突。在传统的配置管理工具中，冲突的解决是基于文件内容的，并且只保存归并的结果，而丢失第二个用户修改的新版本。在 Firefly 中，所有的四个版本（共同的祖先版本，两个用户修改后的私有版本，文件归并之后的最新版本）全部会被系统正确的保存下来，方便日后的回溯，并且可以使用图形化版本树的方式显示出来。请参考示意图 5，6。

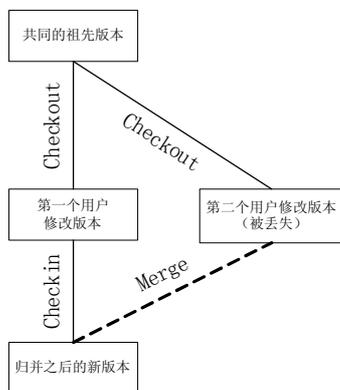


图 5: 传统版本控制工具的归并示意

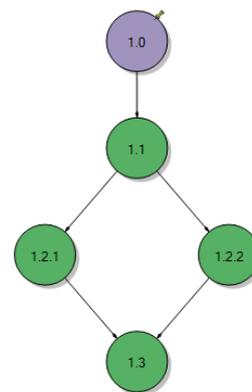


图 6: Firefly 中的版本树

综上所述，相对于基于文件的版本控制工具来讲，Firefly 在处理并行开发，记录并跟踪项目一级的变更，以及处理并发开发等方面有着明显的优势。除此之外，Firefly 还具有如下几大特点：

- 应用了 FleXstream、Java Remoting、Smart Transfer 等先进技术确保了系统执行的速度；
- 基于 TCP/IP, 不依赖于任何共享的网络文件系统, 最大限度的提高了系统的安全性；
- 支持常见的操作系统平台：Windows 的各个版本及多种 UNIX, Linux；

- 支持文件级的安全访问控制；
- 通过 ServerSync 模块支持异地开发；
- 概念清晰，简单易用；
- .....

总之，在软件开发项目中应用 **Firefly**，可以在很大程度上提高整个团队的协作能力和工作效率；不仅能使您完成日常的版本控制工作，还可以完成高级的配置管理功能；使您能够更加从容的面对开发过程中的变化；自如的处理复杂的并行开发问题；为您的软件开发管理水平提升起到巨大的作用。