统崩溃将意味着一个有效的目录入口指向一个i节点，它所列出的磁盘块当前存在于空闲块存储池中并可能很快被再利用。这将导致两个或更多的文件分享同样的磁盘块。这样的结果都是不好的。

日志文件系统则先写一个日志项，列出三个将要完成的动作。然后日志项被写入磁盘（并且为了良好地实施，可能从磁盘读回来验证它的完整性）。只有当日志项已经被写入，不同的操作才可以进行。当所有的操作成功完成后，擦除日志项。如果系统这时崩溃，系统恢复后，文件系统可以通过检查日志来查看是不是有未完成的操作。如果有，可以重新运行所有未完成的操作（这个过程在系统崩溃重复发生时执行多次），直到文件被正确地删除。

为了让日志文件系统工作，被写入日志的操作必须是幂等的，它意味着只要有必要，它们就可以重复执行很多次，并不会带来破坏。像操作“更新位表并标记i节点k或者块n是空闲的”可以重复任意次。同样地，查找一个目录并且删除所有叫foobar的项也是幂等的。在另一方面，把从i节点k新释放的块加入空闲表的末端不是幂等的，因为它们可能已经被释放并存放在那里了。更复杂的操作如“查找空闲块列表并且如果块n不在列表就将块n加入”是幂等的。日志文件系统必须安排它们的数据结构和可写入日志的操作以使它们都是幂等的。在这些条件下，崩溃恢复可以被快速安全地实施。

为了增加可信性，一个文件系统可以引入数据库中原子事务（atomic transaction）的概念。使用这个概念，一组动作可以被界定在开始事务和结束事务操作之间。这样，文件系统就会知道它必须完成所有被界定的操作，或者什么也不做，但是没有其他的选择。

NTFS有一个扩展的日志文件系统，并且它的结构几乎不会因系统崩溃而受到破坏。自1993年NTFS第一次随Windows NT一起发行以来就在不断地发展。Linux上有日志功能的第一个文件系统是ReiserFS，但是因为它和后来标准化的ext2文件系统不相匹配，它的推广受到阻碍。相比之下，ext3——一个不像ReiserFS那么有野心的工程，也具有日志文件功能并且和之前的ext2系统可以共存。

4.3.7 虚拟文件系统

即使在同一台计算机上同一个操作系统下，也会使用很多不同的文件系统。一个Windows可能有一个主要的NTFS文件系统，但是也有继承的FAT-32或者FAT-16驱动，或包含旧的但仍被使用的数据的分区，并且不时地也可能需要一个CD-ROM或者DVD（每一个包含它们特有的文件系统）。Windows通过指定不同的盘符来处理这些不同的文件系统，比如“C:”、“D:”等。当一个进程打开一个文件，盘符是显式或者隐式存在的，所以Windows知道向哪个文件系统传递请求，不需要尝试将不同类型文件系统整合为统一模式。

相比之下，所有现代的UNIX系统做了一个很认真的尝试，即将多种文件系统整合到一个统一的结构中。一个Linux系统可以用ext2作为根文件系统，ext3分区装载在/home下，另一块采用ReiserFS文件系统的硬盘装载在/home下，以及一个ISO 9660的CD-ROM临时装载在/mnt下。从用户的观点来看，那只有一个文件系统层级。它们事实上是多种（不相容的）文件系统，对于用户和进程是不可见的。

但是，多种文件系统的存在，在实际应用中是明确可见的，而且因为先前Sun公司（Kleiman, 1986）所做的工作，绝大多数UNIX操作系统都使用虚拟文件系统（Virtual File System，VFS）概念尝试将多种文件系统统一成一个有序的框架。关键的思想就是抽象出所有文件系统都共有的部分，并且将这部分代码放在单独的一层，该层调用底层的实际文件系统来具体管理数据。大体上的结构在图4-18中有阐述。以下的介绍不是单独针对Linux和FreeBSD或者其他版本的UNIX，而是给出了一种普遍的关于UNIX下文件系统的描述。

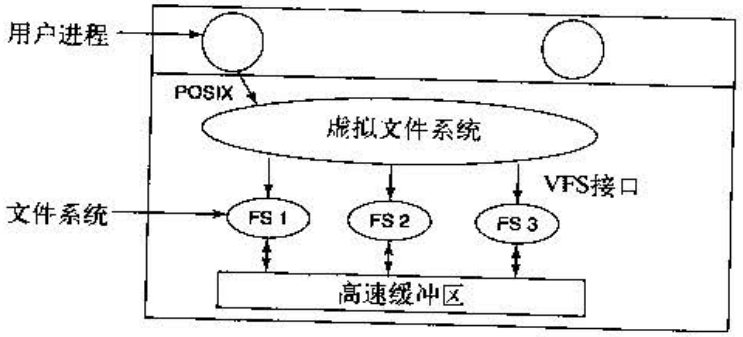

图4-18 虚拟文件系统的位置

所有和文件相关的系统调用在最初的处理上都指向虚拟文件系统。这些来自用户进程的调用，都是标准的POSIX系统调用，比如open、read write和lseek等。因此，虚拟文件系统对用户进程有一个“更高层“接口，它就是著名的POSIX接口。