In chapter 4, the textbook talks about file systems of the Linux system. In particular, the structure of the UNIX file system structure, which is delved into in section 4.14 of the textbook. In 4.14, the relationship between i-nodes and the file system it is contained in. i-nodes are data structures that holds a file’s information, such as access mode (read, write, etc.), the file type, the file size, etc.

There are various types of file systems supported by different operating systems, such as UFS, PCFS, and HSFS. UFS is the traditional BSD-derived UNIX file system, PCFS is a file system to read and write DOS-formatted diskettes, and HSFS is a file system to read CD file systems. A file system is encased in one of many partitions that form the disk drive, and in each file system contains cylinder groups, which then contains the i-nodes among other things like data blocks.

An i-node contains a link count, the number of directory entries pointing to that particular i-node. Only when this link count = 0, can the file by deleted, which is why the function, unlink, does not delete. The link count is stored in an st\_nlink member that is made out of hard links (nlink\_t). Another type of links is the symbolic link that points to actual contents of the file, with a data type of S\_IFLNK. For example, lrwxrwxrwx 1 root 7 (no. of bytes) Sep 25 07:14 lib (filename) -> usr/lib (data).

The i-node contains all sorts of data about the file, but the two most of interest are the filename and the i-node number, which is sort of like an i-node identifier. I-nodes cannot be inferred across file systems, they can only be located within the same file system, which is why the command, ls(1), cannot cross file systems.

Because of the way i-nodes are structured, renaming a file doesn’t require the entire contents of the file to be moved. All that needs to be done is to add a new directory to point to the existing i-node and then unlink the previous old directory entry. For example, renaming /usr/lib/foo to /usr/foo, doesn’t require moving all the contents of foo into /usr. This is how the mv(1) command operates.