There is already a file named *ch2* in the *work* directory. When **cp** asks, answer **n** to prevent copying *ch2*. Answering **y** would overwrite the old *ch2*.

As you saw in the section "Relative pathnames up" in Chapter 3, the shorthand form **.** puts the copy in the working directory, and **..** puts it in the parent directory. For example, the following puts the copies into the working directory:

```
$ cp ../john/ch[1-3] .
```

cp can also copy entire directory trees. Use the option **-R**, for "recursive." There are two arguments after the option: the pathname of the top-level directory you want to copy from, and the pathname of the place where you want the top level of the copy to be. As an example, let's say that a new employee, Asha, has joined John and Carol. She needs a copy of John's *work* directory in her own home directory. See the filesystem diagram in Figure 3-1. Her home directory is */users/asha*. If Asha's own *work* directory doesn't exist yet (important!), she could type the following commands:

```
$ cd /users
$ cp -R john/work asha/work
```

Or, from her home directory, she could have typed "**cp -R ../john/work work**". Either way, she'd now have a new subdirectory */users/asha/work* with a copy of all files and subdirectories from */users/john/work*.

If you give **cp -R** the wrong pathnames, it can copy a directory tree into itself—running forever until your filesystem fills up!

If the copy seems to be taking a long time, stop **cp** with CTRL-Z, then explore the filesystem (**ls -RF** is handy for this). If all's okay, you can resume the copying by putting the **cp** job in the background (with **bg**) so it can finish its slow work. Otherwise, kill **cp** and do some cleanup—probably with **rm -r**, which we mention in the section "rmdir" later in this chapter. (See the section "Running a Command in the Background" and the section "Cancelling a Process" in Chapter 7.)
