IEEE Standards Interpretation for IEEE Std 1003.1™-1990 IEEE Standard for Information Technology--Portable Operating System Interfaces (POSIX®)

Interpretation Request #79
Topic: unlink **Relevant Clauses:** 5.5.1

I would like to get IEEE's interpretation of the compliance of Vendor XXX implementation of the unlink function on the YYY operating system. In particular, the unlink function fails to remove the last directory entry for a file if that file is an executable that some process is running. The following two small C programs can be used to demonstrate the unlink failure on YYY O/S. This first program should be compiled and run with an argument which specifies the length of time this program should sleep. /* Running executable (sleeping). */ /* Run with an integer argument to specify sleep time. */ #include #include #include int main(int argc, char *argv[]) { int seconds = 0; if (argc != 2) { fprintf(stderr, "Supply an integer for the sleep argument.\n"); exit(EXIT_FAILURE); } seconds = atoi(argv[1]); if (!seconds) { fprintf(stderr, "Supply an integer for the sleep argument.\n"); exit(EXIT_FAILURE); } sleep(seconds); exit(EXIT_SUCCESS); }

This second program will attempt to unlink the running executable which was generated from the previous code. This program needs an argument which specifies the name of the file which is to be unlinked. #include #include #include int main(int argc, char *argv[]) { if (argc != 2) { fprintf(stderr, "Supply a filename for the unlink argument.\n"); exit(EXIT_FAILURE); } if (unlink(argv[1])) { fprintf(stderr, "Error unlinking %s\n", argv[1]); perror(NULL); exit(EXIT_FAILURE); } exit(EXIT_SUCCESS); } Local testing has shown that the second program will fail to unlink the running executable on YYY O/S systems. The unlink succeeds on all other UNIX systems that I have tested.

Interpretation Response
The implementation is conforming if the errno returned in this case is EBUSY or another error not listed in 5.5.1.4, and the behavior is described in the conformance documentation.

Rationale for Interpretation

Clause 2.4 allows any function to fail for "additional errors." The error number generated can be the same as one of those listed if the corrective action taken by the application is identical to the condition described in the standard. Since there is nothing a conforming application can do for an EBUSY error from unlink(), it is acceptable to return this error when the file itself is busy instead of the directory.