fession is much older than programming, it might provide additional, beneficial guidelines to program design. Second, the similarities between the two fields imply that training in rhetoric might well be excellent preparation for programming, tapping a source of programmers long ignored because of historical concentration on the importance of quantitative background and training.

The contention that training in rhetoric might be excellent preparation for programming is informally supported by the common impression that a disproportionate number of applications programmers are English majors who could not find editing or writing jobs. More formal confirmation could come from an empirical study comparing training time, performance, and other relevant variables for programmers with a good backgroun in composition and against those without such a background.

Before conducting the empirical study described above, one can make three contentions about the connection between programming and rhetoric that stand on their own merits. The first is that the two activities appear to be fundamentally very similar. The second is that training in rhetoric can provide would-be programmers with practice in modularization, organization, and clarity of expression that can carry over into programming. The third is that such training will also improve would-be programmers' skills in oral and written communication. Such skills will be as important to them in the long run as technical knowledge.

References

John A. Lehman is an assistant professor in the Department of Management Sciences, School of Management, University of Minnesota. His teaching specialty is programming design, and his research interests are software engineering and computer graphics. Lehman received his BA and MA in Chinese and his MBA and PhD in computer information systems, all from the University of Michigan, Ann Arbor.

The author's address is Dept. of Management Sciences, School of Management, University of Minnesota, MN 55455.