DISCUSSION OF JACOBS PAPER

Jeanne C. Adams
National Center for Atmospheric Research

The field of computing has grown considerably in recent years and the tasks performed on computers encompass more and more variety in scientific, industrial and business applications. The programming task itself must be defined in very broad terms if one is to cover all the activities performed in all areas where computers are used today.

A personnel manager, in selecting among applicants for a programmer position, must take into account the kind of programming the applicant will be expected to do. Consider that there are systems programmers, data analysts, mathematical programmers and social science programmers. Programmers may use large systems or mini computers. They may be programmer professionals and computer scientists or they may be scientists who use programming skill merely as one among many research tools. They may need to consider the total machine environment, as does a systems programmer, or they may need only learn a high level language in a particular kind of research application. Because of the complexity of the task and the variety of applications it is difficult to define what a programmer does.

In his paper, Mr. Jacobs defines the programming task in very broad terms. He recognizes that current tests for personnel selection are not satisfactory and suggests that certain tests of cognitive capabilities might prove better instruments for recognizing potential programming skills. However, the predictor variables did not correlate significantly with the criterion variable, success in programming training, and his hypotheses were not supported.

His sample population (the size of which is not specified) consisted of students in university courses in computer programming, only one-third of whom were taking the courses as a requirement and we can only guess what proportion of them are actually planning careers in this area. Two-thirds were taking the courses because they were generally "curious" about the subject. I doubt that cognitive factors alone should be expected to predict differences in performance in this context. The students' attitudes and goals in the course must be considered. Two-thirds of these students may not apply themselves wholeheartedly because the course is not

central to their career goals. They are only curious. In any job, low performance may be due to lack of motivation or concern for the goals central to the task. Thus the non-cognitive factors may outweigh the cognitive factors in determining success.

I understand the term "programmer trainee" to refer to someone who has been hired for a probationary period during which he or she will be trained to do programming. A person in such a position has usually made at least a minimal commitment to try to perform well the task of programming. It is not clear to me that even one-third of this student group could be considered similar to "programmer trainee."

The final examination used as the indicator of programming performance emphasized only a small part of the total skills required for adequate computer programming. The student was asked to recognize errors in syntax and trace some very simple arithmetic algorithms that were already written. Competence in designing logical definitions and ability to express one's ideas in the language were not tested.

Although it is convenient and usually less expensive to study students in a university course, I would suggest that we might learn more about the factors making for success in computer training if we studied persons who actually become programmers, including excellent programmers. Even if that is done, the problem of finding adequate predictors may not be simple.

I recall that a number of years ago at NCAR an attempt was made to identify capabilities and attitudes which, when manifested by job applicants, predicted that applicants would become excellent, rather than simply adequate or mediocre, programmers. Experienced programmers working at NCAR were rated according to their performance and were given a battery of tests, most of which emphasized mathematical ability and logical thinking. The discouraging result of this unpublished study was that the excellent, the most highly-rated, programmers showed a very wide range of test scores.

Of course that study suffered from the fault of being ex post facto. It was probably not safe to assume that whatever factors made for excellence in programming by the experienced person had been present when that person first applied for a programmer trainee position. So even if the test scores of the "excellent" programmers had clustered in the higher ranges, those tests still might not have turned out to be good predictors.

The most desirable way to identify factors making for programming success, and to find tests which will serve as predictors, would be to follow persons through the series of steps from when they first seek training and/or employment in programming to their eventual "mature" performance as full-time programmers. That might cover a five year period.

Further, it would be valuable to study not just cognitive factors but also affective factors and goal direction; that is, the attitudes and career commitments of would-be programmers might well turn out to be more important than cognitive style or capability. An example of a promising attempt to identify affective factors is the paper read before the 1970 meeting of this group by Lucy Zaborenko in which she described the use of projective tests to assess programmers' attitudes toward computers.(1)

Finally, such an "ideal" study should include a differentiation among, and a comparison of, the factors which make for success in at least three possible types of programmer roles:

1. The applications programmer whose work is primarily in support of scientific research in the physical or behavioral sciences. Such a programmer might be characterized as "scientific problem-oriented."

2. The programmer whose work is "service-oriented," that is, in support of a variety of users including business and commercial. Such a programmer is concerned with the most efficient code, the design of the best program for a given computer system.

3. The systems programmer whose work is to design operating systems. Such a programmer is concerned with the total hardware and software environment in which specific computing problems are solved.

I suspect that the relative importance of cognitive, affective and goal commitment factors may well be quite different among these three types of programmers; that is, success in such different roles may require quite different cognitive capabilities, affective attitudes and goals.

But, of course, such an ideal study would be extremely expensive of effort and resources. And a "growth" study of the sort I have suggested is subject to great hazards of attrition, especially through the loss of cases and even of research personnel.

REFERENCES

1. Zaborenko, Lucy, George F. Badger, Jr. and Ellen B. Williams. "TABRA: A Projective Test for Computer Personnel Research. Preliminary Report." Proceedings of the Eighth Annual Computer Personnel Research Conference, June 22-23, 1970, pp. 92-107