Sex Bias in Educational Software: The Effect of Designers' Stereotypes on the Software They Design

Charles Huff & Joel Cooper

Princeton University

We are indebted to Bob Kraut, Steve Hanson, and an anonymous reviewer for helpful comments on an earlier version of this manuscript. Reprint requests should be sent to: Charles Huff, Psychology Department, St. Olaf College, Northfield MN 55057 USA.

This paper appeared in
Journal of Applied Social Psychology, 1987*, 17,* 519—532.

Abstract

Why is the computer more alluring to boys than it is to girls? One answer to this question is drawn from a social psychological model of human interaction. Social psychological research indicates that the expectations an individual has about another person can shape his or her interaction with that person. We hypothesized that, in a similar manner, the expectations software designers hold about the users of the software they design are central in determining the way the software they design interacts with the user. In order to test this notion, we had 43 educators with programming experience design software for either boys, girls, or students, and found that programs for girls were classifiable as "learning tools" while programs for both boys and students were most like "games." These differences occur as a function of the designers' expectations of the characteristics of potential users of the program, and result in sex stereotyped software. We conclude that it is not the computer, or even the software, that is at the root of the sex bias in software, but the expectations and stereotypes of the designers of the software.

Sex Bias in Educational Software:
The Effect of Designers' Stereotypes
on the Software They Design

The world of computers and computing technology is mostly inhabited by men. Though the percentage of women in computer-related jobs is increasing, this increase is mainly caused by an increase in low-status, low-pay computing jobs that have no access to higher levels of management or to better paying positions (Kraft & Dubnoff, 1986). In fact, these positions are mainly extensions of already female dominated jobs, such as the "upgrading" of secretarial positions to "word-processing" positions (Iacono & Kling, 1984; Gutek & Bikson, 1985).

Why are women so under-represented in computing? In addition to the structural barriers that keep women from being hired (Rosen & Jerdee, 1974), it has been shown that young girls have less access to computers, both in school and at home, and therefore less of a chance to become skilled in using them (Hess & Miura, 1985). To compound this difficulty, most of the educational software available today is presented in a format that, while motivating and exciting for boys, probably discourages the girls who use it (Lepper & Malone, 1985). Thus, even if a girl has access to a computer, she may well have difficulty enjoying an interaction with it, and may be discouraged from further attempts to use it.

Why is the computer more alluring to boys than it is to girls? This is really a question in human-computer interaction: why do some people find their interaction with the computer more pleasurable than do others? In an attempt to answer this question, we turned to social psychological analyses of human-human interaction. Social psychological analyses have been conducted of the ways people interact with each other using the computer as a messaging system (Kiesler, Zubrow, & Moses, 1985; Kiesler, Siegel, & McGuire, 1984). However, we propose that social psychological research and theory can, in addition, help us understand the ways people interact with the computer itself, and help uncover the sources of the computer's approach to people. Both software and hardware are, after all, produced by people; and it is those people who determine the design of the computers with whom both we and our children interact.

Social psychological theory and experimentation make evident the fact that a person's expectations, stereotypes, and prototypes of other people will determine the way he or she interacts with those people. For instance, subtle changes in the approach of an interviewer can determine the success or failure of an interviewee (Snyder & Swann, 1978; Word, Zanna, & Cooper, 1973) and these changes in the interviewer's actions are probably a function of the expectations and stereotypes the interviewer has about the prospective applicant (Darley & Fazio, 1980). Word, Zanna, & Cooper (1974) documented the strategies interviewers used in interacting with black and white applicants, and found that white applicants were given different non-verbal cues than were black applicants. They also found that if white applicants were treated in the same way that black applicants were, the whites' performance in the interview suffered. Other research (see Fiske & Taylor, 1984) also indicates that the expectations that one person holds about another can affect both the actual performance of that other and the way in which the performance is perceived.

The relevance of this work to software design is immediately apparent: the expectations of the software designer are central in determining the design of the program. And it is often the program, and the program's approach to the user, that determines the success or failure of the user in his or her interaction with the computer. The logical outcome of this line of reasoning is that it is not the computer, or even the software, that is at the root of the sex bias in software, but the expectations and stereotypes of the designers of the software. What are these expectations like, and what effect do they have on the design of educational programs? When designers are asked to write a program for school children, for whom is the software written, and what prototype of the user comes to their minds?

In an attempt to answer this question, we designed an experiment to determine the sorts of software designers would write for both boys and girls, and compared these programs to ones written for "students." What sorts of differences among these programs might we expect? Previous research in sexual stereotyping (Broverman, Vogel, Broverman, Clarkson, & Rosenkrantz, 1972) has indicated that the stereotype of the "healthy" male is quite different from that of the "healthy" female; more disturbingly, the stereotype of the "healthy" person is quite different from that of the healthy woman, but almost identical to that of the healthy man. This indicates that when asked to think of a "healthy" person, respondents were probably thinking of a healthy man. Similarly, when we ask people to program for "students" they may well direct their program toward male students, assuming either implicitly or explicitly that the programs' users will be boys. This will produce programs that are intended to appeal only to males.

Thus, we expect that programs written for students will be most like those written for boys, but that programs written for girls will resemble neither. Given the composition of software on the market today, we expect that boys' (and therefore students') programs will be action oriented games, though we do not know what sort of programs will be produced for girls.

Method

Characteristics of the Program Designers

Program designs were obtained from 43 educators in New Jersey. 34 were female and 9 were male. They averaged 15.7 years experience in teaching, and each knew (on average) 2.2 programming languages. Though predominantly teachers, there were also librarians and school administrators in the group. They taught a large range of grades (from 1st grade to college), with the median being 6th grade.

Participants were solicited for the study using several methods. Nine were taking a class at Rutgers University in teaching computing, 20 responded to a questionnaire sent to members of CLUES, a New Jersey organization of educators interested in computing. 14 were taking a computers and education class in a local school district. The 14 participants in this class were given photo copies of the questionnaire sent the teacher, and were thus all assigned to one condition.[1]

The Experimental Design

Participants were asked to design a program to teach 7th grade pupils to use commas correctly. Some were asked to design a program for 7th grade girls, others to design a program for 7th grade boys, and others to design a program for 7th grade students. We thus established a one factor design with three levels: programs designed for 7th grade boys, girls, or students.

Obtaining the Designs

Each participant was given the following instructions about their task:

> Imagine you are teaching a class of 7th grade (Boys / Girls / Students). A subject with which the (Boys / Girls / Students) seem to be having some trouble is the proper use of commas. Please design a computer program that will help them acquire this skill. You are only designing the program, and the design you create will be used by a programmer to produce a program your (Boys / Girls / Students) can use. Do not concern yourself with hardware or software limitations, just design the best program you can. To help you in this task, on the next several pages are some questions about the program you will be designing. By the time you have answered these questions with about a paragraph apiece, you should have developed a program.

The set of questions we gave the designers are presented in Table 1. Participants wrote free-form replies to each question, and, when they had finished answering all questions wrote a short summary of their program.

Insert Table 1 about here

Obtaining the Program Ratings

Designers, upon finishing their program design, were presented with 23 scales to use in rating the program they had just created. They were asked not to modify their program design, but to rate it in its present form. The scales asked the designers about: how competitive the program encouraged the child to be, the percentage of time a child would be actively doing things (rather than watching things), the necessity of quick reflexes and dexterity, the type of feedback presented, the amount of action on the screen, the vocabulary required of the user, the length of a session and the amount of time needed to learn the program, whether the program encouraged aggressiveness, the control over feedback and levels of difficulty given the user, and the presence and importance of sound, color, and graphics. We also asked them to indicate whether their program presented itself as mostly a game or as mostly a learning tool. When they had finished rating their program, they also provided us with demographic information about themselves.

---

[1] When a discriminant analysis is performed without including these 14 subjects, the first function is substantially similar to that computed when these designers' ratings are pooled with the other, randomly assigned designers. The first function is marginally significant (χ^2 (6, N =27) = 16.14, p = .09) and the relative positions of the group means on the function are retained. Given this result (and the replication of the function from independent ratings), we think it is safe to assume that the arbitrary assignment of these 14 designers to the "design for girls" condition did not produce the effects we report.

Obtaining Independent ratings

Three graduate students were paid to rate the program descriptions. Each student rated all 43 programs. Those scales that were found reliable (a > .6) were used to compute an "independent" program score for that scale. This score consisted of the average rating of the three students. The scores were used to provide a check on the pattern of data resulting from the designers' ratings of their own programs.

Results

Discriminant Analysis

Our design involves multiple dependent variables (see Table 2) whose purpose is to help us discover differences among the three experimental groups. Discriminant Analysis (Lachenbruch, 1975; Klecka, 1980) allows us to determine which subset of the variables, when combined in a weighted sum, best exposes the differences among the three groups. When only two groups are involved, Discriminant Analysis is conceptually similar to Multiple Regression. A function (or dimension) is computed that is a linear combination of variables. This function is set up to maximize the differences between the groups. The weights of the variables on the function are interpreted just like weights in Multiple Regression. Larger standardized weights imply a larger contribution by that variable, and the sign of the weight indicates the direction of correlation of the variable with the function. When more than two groups are involved, Discriminant Analysis can produce k—1 functions (where k is the number of groups). Thus, the groups are placed in a k—1 dimensional space, with each dimension attempting to differentiate among the groups.

Results from Designers' Ratings

All 23 dependent variables were included in the discriminant analysis. Four of the designers provided missing data and were excluded from the analysis, leaving a total N of 39. The Mahalanobis D^2 (a measure of distances between groups), was used to measure the success of a variable in discriminating among the groups (see Lachenbruch, 1975; Mahalanobis, 1963). If a variable significantly increased D^2 between 2 groups, it was added to the function. If a variable already in the function failed to contribute to the D^2 between any two groups, it was removed from the function. This stepping procedure was continued until there were no more additions or deletions to be done given the criterion.

Since there are three groups, it is possible to compute two functions to describe the differences among them. The first function discriminated significantly among the groups (□^{2} (6, N =39) = 17.26, p < .01) and differences among the groups accounted for 47% of the variance on this function. Figure 1 presents a histogram detailing the placement of each program on this dimension.

Insert Figure 1 about here

There are two statistics we must consider in attempting to interpret this function. The first is the weight of each discriminating variable on the function. These are interpreted in the same way Beta weights are interpreted in Multiple Regression. Equally important is the correlation between each variable (even those not in the solution) and the function. A variable may not have been included in the solution because it shared most of its discriminating power with a variable already in the solution. Even though that variable is not in the function, if it has a high correlation with the function it can help us interpret it. Table 2 contains the relevant information for the analysis of both the designers' ratings and the independent ratings.

Insert Table 2 about here

It can be seen from this table that as one moves to the left on the function depicted in Figure 1, the programs: require more eye to hand coordination, engender more competition in the user, and present themselves more as games. If one moves to the right on the function, programs: take more time to use, give the user more control over the type of feedback presented, give less control over the difficulty levels, and present themselves more as learning tools. Color is more important in programs on the right side of the function, and these programs also have louder sound effects. It seems that programs on the left of the function are "games," with eye to hand coordination and competition most important, while programs on the right of the function are "learning tools," with preset difficulty levels and long sessions. The one anomaly in this interpretation is that the tools have louder sound effects than do the games. We think this can easily be resolved. Designers often expressed concern that sound effects would disrupt the classroom. They seemed to take two paths in solving this problem. The "games" were usually given no sound, and though the learning tools often had melodies played by the computer, the sound was seldom turned up over half volume. Thus, the games tended to have no sound (which was coded as zero volume) while the learning tools had moderately loud melodies playing.

The outcome of most interest to this study is the placement of the groups on this discriminant function. Which groups are different from which others and where do they fall on the "game-tool" dimension? From the 2 function solution, it is possible to compute F values representing the difference between the means (centroids in 2 or more dimensions) of each pair of groups. We find that the girl programs are significantly different from both the boy and the student programs (F (7, 30) = 2.54, p < .05, F (7, 30) = 3.66, p < .01, respectively). However, the boy programs are only marginally different from the student programs (F (7, 30) = 2.15, p = .069). In addition, most of the difference between the boy and the student programs occurs on the second, marginally significant, function. On the "game-tool" dimension, the two groups are almost identical (M = -.99 and -.96 for students and boys, respectively). Thus, both boy and student programs are similar in being game oriented, while girl programs are learning tools.

Though with 3 groups it is possible to define 2 functions, the second function was only marginally significant (p = .058). This second function apparently differentiates student programs from boy and girl programs. The weights and correlations indicate that as one moves toward the student pole on this function, the programs are more like games, and sound and color become more important (see Table 2). As one moves toward the boy/girl end of the dimension, the programs become more like tools and require more typing skill. Why this function appears in this form is unclear, and since it does not re-occur in the independent ratings, any substantive interpretation would probably be premature. It may be that this dimension is a reflection of the extreme "game-likeness" of the student programs.

It is interesting to note that if all the male designers are dropped from the analysis, the result remains the same. A game-tool function emerges as the first function (χ^2(6, N =29) = 18.72, p < .01), and the second function is non-significant (p > .1). Again, student and boy programs are on the game end of the dimension, while girl programs are on the tool end.

Results from Independent Ratings

Before we can analyze the independent ratings, we must first determine if the raters agreed with each other regarding the attributes of the programs. We computed an alpha coefficient for each of the 23 scales and only used scales with a reliability of .6 or greater. Since the written descriptions of the programs were sometimes sketchy, it is not surprising that only 15 of the scales met the criterion for inclusion in the discriminant analysis. These scales can be seen in Table 2. In addition, the volume of the program was rated using 2 different scales: one to indicate presence or absence of sound effects, and the second to determine the volume of sound effect, if present.

Discriminant analysis of these 15 scales produced only one discriminant function (again, the second function was not significant, p > .65). The first function discriminated significantly among the groups (χ^2 (3, N = 43) = 13.09, p < .01) with the differences between the groups accounting for 27% of the variation on the function. This function was quite similar to the "game-tool" function found in the designers' ratings. Programs to the left of the dimension: required more eye to hand coordination, more quick reflexes, presented more action on the screen, engendered more aggression in the user, and presented themselves more as games.

Only girl and student programs were significantly different from each other (F (4, 37) = 3.37, p < .05) in this analysis. Boy programs were not different from girl or student programs (p's > .3). However, the relative position of the groups on the dimension was preserved (see Figure 1). Student programs are the most game—like, boy programs are in the middle, and girl programs are on the "learning tool" side of the function.

It is interesting to note that in this analysis, the anomalous effect of sound volume is replaced with an effect for the presence of sound effects—girl programs are more likely to have sound, but volume of sound has no relationship to this dimension. This confirms our earlier interpretation of the relationship of volume to the game-tool dimension. In addition, some of the expected correlations that did not appear in the designers' ratings are present in the independent ratings. According to the independent ratings, boy and student programs require more eye-to-hand coordination, quicker reflexes, and contain more action on the screen (these relationships did not appear in the designers' ratings). The relationships among the variables in the independent ratings appear to be more consistent than that found in the designers' ratings. In general, the two game-tool functions (from designers and independent raters) are similar, and the independent raters' solution displays some results we had expected, but not found, in the designers' ratings. This seems a reasonable confirmation that the game-tool function found in the designers' ratings actually describes differences among the programs they designed.

Discussion

There does, indeed, seem to be a difference in software written for boys and that written for girls. That this difference exists is not surprising. There is evidence to indicate that boys, in fact, prefer games, and that girls prefer "learning tools" (Lepper & Malone, 1983; Cooper, Hall, & Huff, 1985). It would not be extraordinary for teachers to notice these preferences and to take them into account when designing software.

The finding that should concern us is that programs written for students are written, it seems, with only boys in mind. Given that most young computer users are male (Hess & Miura, 1985; Hawkins, 1985), our designers may have assumed that their users would be male and intentionally set out to please the male majority. This explanation seems unlikely to us, given that most of our designers were female, and often expressed concern regarding the sex bias in educational software. It is likely that our designers, when asked to design a program for students, assumed without reflection that "students" are male. That is, they may have been simply using "male" as the default value of "student" (Hofstadter, 1982). This is quite similar to sex-stereotyping results in other fields where it does appear that people thinking of a person were in fact thinking of a male (Broverman, et al., 1972; Phillips & Gilroy, 1985).

There is one obvious implication of the "male bias" we have documented in software design. Educational software may, by and large, be designed to appeal to boys, without consideration of the effect on girls' motivation to use them or on girls' educational profit from them. This certainly cannot be a good thing. In order to document this assertion, it would be necessary to rate some of the educational software on the market in regard to this "game-tool" dimension. It is our expectation that a great many of these programs will fall on the "game" end of the dimension - programs designed with boys in mind.

These findings of sex bias in educational software should be interpreted with caution. Our findings clearly indicate that different programs are designed for boys than are designed for girls, and that programs designed for students are similar to those designed for boys. What our data do not tell us, however, is how educational software should be designed. At most, we may have a handle on one thing we ought to avoid in educational software. On the basis of the present data, we cannot recommend that software be written to fall at any point on this "game-tool" dimension. We may need software that is both a game and a tool, thus making the dimension irrelevant. These data do give us insight into how educational software is designed, but they have little positive to say about how it ought to be designed. We first need to know if programs designed in this manner have a deleterious effect on their intended users.

We have collected some data (Cooper, Hall, & Huff, 1985) that indicate that children using software designed for the opposite sex are more anxious after they interact with the program, and that that anxiety leads to lowered scores in the subject the program was intended to teach. However, this only occurs if the children are using the program in public, that is, in a computer lab with other children present. When the programs are used privately, these differences do not emerge. These data, then, indicate that it is not only the software that may be at fault, but the situation in which the software is presented. More research (and theory) is needed if we expect to identify the parameters that lead to sex bias in software and to assess its impact on the children who use it (Lepper, 1985).

Another conclusion we may draw from this work is that social psychological theories of human interaction may well be useful in work in human-computer interaction. This has proven true in this instance, and should encourage those who are interested in social aspects of computing. Programmers' stereotypes of the users of their software will most likely affect the design of their programs regardless of the field in which the programs are designed, so we expect these findings will be applicable to areas other than education. In addition, there are other areas of social psychological theory that may have relevance for human-computer interaction. Social psychologists investigate the causal explanations people give for their environment (Wong & Weiner, 1981), the way people react to attempts of control by others (Brehm, 1972), the ways people form groups to solve problems (Stiener, 1976), and a host of other issues that should provide insight into how people interact with, react to, and attempt to master, use, or co-operate with computers.

References

Brehm, J. W. (1972).. *Responses to loss of freedom: A theory of psychological reactance.* Morristown, NJ: General Learning Press.

Broverman, I. K., Vogel, S. R., Broverman, D. M., Clarkson, F. E., & Rosenkrantz, P. S. (1972). Sex—role stereotypes: A current appraisal. *Journal of Social Issues, 28, 59—78.*

Cooper, J., Hall, J., & Huff, C. W. (1985). [Effects of using sex stereotyped software]. Unpublished raw data.

Darley J. M., & Fazio, R. H. (1980). Expectancy confirmation processes arising in the social interaction sequence. *American psychologist*, *35*, 867 - 881.

Fiske, S. T., & Taylor, S. E. (1984). *Social cognition*. Reading, MA: Addison-Wesley.

Gutek, B. A., & Bikson, T. K. (1985). Differential experiences of men and women in computerized offices. *Sex Roles, 13,* 123—136.

Hawkins, J. (1985). Computers and girls: Rethinking the issues. *Sex Roles, 13,* 165—180.

Hess, B., & Miura, I. (1985). Gender differences in enrollment in computer camps and classes. *Sex Roles, 13,* 193—204.

Hofstadter, D. R. (1982). Default assumptions and their effect on writing and thinking. *Scientific American, 249,* 18—36.

Iacono, S., & Kling, R. (1984). Office routine: The automated pink collar. *IEEE Spectrum. 21,* 73—76.

Kiesler, S., Siegel, J., & McGuire, T. (1984). Social psychological aspects of computer mediated communication. *American Psychologist, 39,* 1123—1134.

Kiesler, S., Zubrow, D., & Moses, A. M. (1985). Afffect in computer mediated communication: An experiment in synchronous terminal–toterminal discussion. *Human Computer Interaction, 1,* 77—104.

Klecka, W. R. (1980). *Discriminant analysis.* Beverly Hills, CA: Sage Publications.

Kraft, P., & Dubnoff, S. (1986). Job context, fragmentation, and control in computer software work. *Industrial Relations, 25,* 184—196.

Lachenbruch, P. A. (1975). *Discriminant Analysis.* New York: Hafner.

Lepper, M. R. (1985). Microcomputers in education: Motivational and social issues. *American Psychologist, 40,* 1—18.

Lepper, M. R., & Malone, T. W. (1987). Intrinsic motivation and instructional effectiveness in computer–based education. In R. E. Snow, & M. C. Farr (Eds.), *Aptitude, learning, and instruction: III Conative and affective process analyses.* pp. 255—286. Hillsdale, NJ: Erlbaum.

Mahalanobis, P. C. (1963). On the generalized distance in statistics. *Proceedings of the National Institute of Science, India, 12,* 49—55.

Phillips, R. D., & Gilroy, F. D. (1985). Sex–role stereotypes and clinical judgments of mental health: The Brovermans' findings reexamined. *Sex Roles, 12,* 149—193.

Rosen, B., & Jerdee, T. H. (1974). Influence of sex role stereotypes on personnel decisions. *Journal of Applied Psychology, 59,* 9—14.

Snyder, M., & Swann, W. B. (1978b). Behavioral confirmation in social interaction: From social perception to social reality. *Journal of Experimental Social Psychology. 14,* 148-162.

Stiener, I. D. (1976). Task performing groups. In J. Thibaut, J. Spence, & R. Carson (Eds.), *Contemporary ternds in social psychology,* (pp. 393—422). Morristown, NJ: General Learning Press.

Wong, P. T. P., & Weiner, B. (1981). When people ask "why" questions, and the heuristics of attributional search. *Journal of personality and social psychology*, *40*, 650-663.

Word, C. O., Zanna, M. P., & Cooper, J. (1973). The nonverbal mediation of self–fulfilling prophecies in interracial interaction. *Journal of Experimental Social Psychology, 10,* 109—120.

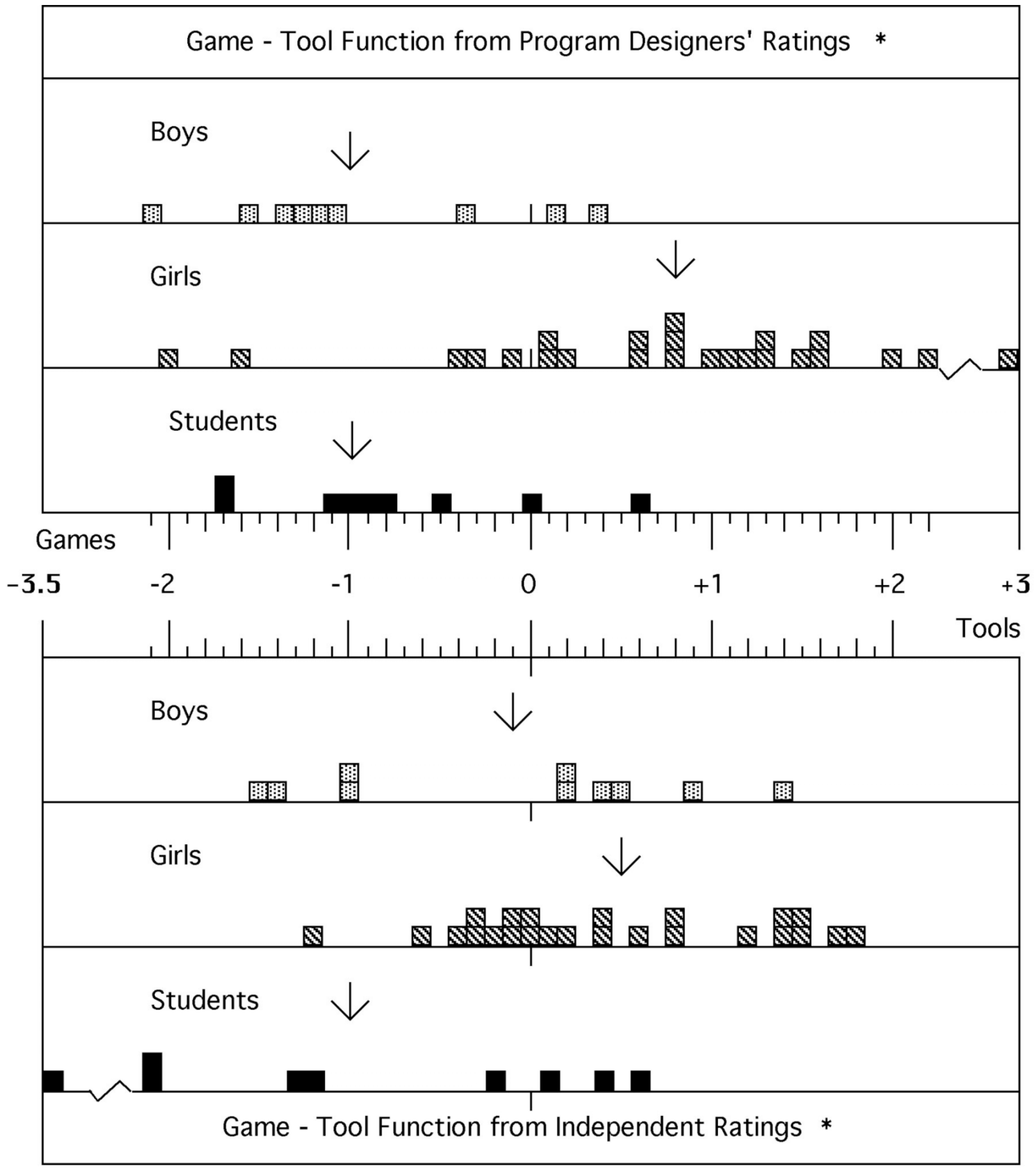
Table 1.
Prompt questions to aid program design.

1) What will the theme of the program be (a trip, a conversation, a quiz, a game)?
2) How will the child interact with the program (through typed commands, a menu, a joystick, etc.)?
3) What will the child do while using the program (converse through typing, suggest strategies to the computer, etc.)?
4) How will the child learn that a response was correct or not? (what kind of feedback will be presented or available)?
5) What kind of sound, color, or action will be incorporated in the program?
6) What will the child see on the screen while using the computer? Describe, point by point, a short interaction that might take place between the computer and the child in a session.
7) How will the child get directions to use the program?
8) What levels of difficulty will be available and how will the program choose among them?
9) Will the program give any printed output? If so, describe it.

Table 2. Item Weights and Pooled Within Groups Item-to-Function Correlations

	Designer's Ratings				Independent Ratings	
	Weight in Function		Correlation		Weight in Function	Correlation
Item Name	Function 1	Function 2	Function 1	Function2	Function 1	Function 2
Volume (Sound)	.953	-.204	.362	-.177		.068
Session Length	.548	-.318	.313	-.077		
Competitiveness			-.263	-.105		-.162
Importance of Sound			.213	-.206		.053
Tool vs. Game	-.722	.577	-.172	-.058		-.245
Control of Feedback	.907	.977	.416	.476		
Importance of Color	.230	-1.049	.145	-.456		
Typing Skill			.069	.250		.085
Coordination	-.449	.215	.091	.188		-.379
Action on Screen			.072	-.153	-1.415	-.387
Aggressiveness			.106	.115		.270
Control of Difficulty	-.721	-.106	.014	.102		-.005
Reflexes			.071	.092		-.260
Type of Feedback			.033	.076	.648	.422
Importance of Graphics			.043	-.042	1.052	.103
Presence of Sound (rated by Independent only)					.521	.135

Figure 1: The Game-Tool Dimension for Designers’ and Independent Raters’ Ratings.



* Arrows indicate group means.   Each square represents one program.