Position Statement for the
Workshop on Strategic Directions in Computing Research
Working Group on Human Computer Interaction

Brad A. Myers

Human Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213-3891
(412) 268-5150
bam@a.gp.cs.cmu.edu
http://www.cs.cmu.edu/~bam

Computers have already saturated businesses, and an estimated 80% of all salaried workers will work at computers by the year 2000. Computers are also in 34% of all homes in the US, including 38% of all families with children under 18. Increasingly, these computers are used for recreation, education and information gathering, in addition to the more conventional business tasks of word processing, financial calculations and databases. Over eight in ten families planning a PC purchase in the next year cite "children's educational use" as the reason for the purchase.[1]

Programming for Everyone

One important aspect of the user interface that has been neglected is the *programmability* of the computers. A primary reason that personal computers became popular in workplaces is due to spreadsheets, and spreadsheets are useful because users can write formulas and macros to customize them to perform calculations relevant to their particular needs. This is a form of programming. Many other applications can also be customized, extended and scripted, including Microsoft Word with Visual Basic, essentially all CAD programs, and Unix shells. Many educators argue that students learn best when *creating*, so some of the best educational software allows the student to construct simulations and environments, often through a form of programming. Part of the excitement about Java and JavaScript stems from their ability to let users write programs for the World-Wide-Web.

However, there has been a woeful lack of research into how to make programming effective and useful for everyone. Most of today's programming languages are quite poor from a human-interface perspective. For example, Java and JavaScript are clearly designed for professional programmers, and contain syntax and semantics that are well known to be hard for novices to understand. Visual Basic and the scripting languages in spreadsheets and CAD programs are based on conventional programming languages. The result is that the vast percentage of computer users actually

1. *EDP Weekly*, by Computer Age, Vol. 36, no. 41, Oct. 23, 1995. For these an many other statistics about computers, see http://www.cs.cmu.edu/~bam/numbers.html.

cannot customize or extend their computer programs. Traditional programming language research is moving in the direction of making languages more mathematical and formal, so that programs can be more easily proven correct. However, this will probably make the languages *harder* for most people to understand and create programs. Therefore, it is up to the HCI researchers to develop appropriate languages and paradigms that will bring programming to the general public. Approaches such as demonstrational programming, visual programming, automatic creation of programs (for example, by “intelligent agents”), and better design for languages with HCI principles in mind, are promising directions for helping with this effort.

Heuristics in the Interface

There are many forces that are moving interfaces away from the conventional “direct manipulation” style, where the user is required to directly control everything about the interaction, to a more cooperative style where the computer facilitates the interaction. Examples in today’s interfaces include Microsoft “Wizards,” “Smart” features in text and graphics editors and spreadsheets, and Intelligent Agents. Furthermore, gesture, handwriting, and speech recognition are all now becoming practical as processors finally get fast enough. All of these interfaces share the feature that the computer can mis-interpret the user’s intentions and do the wrong thing. However, there has been very little research on feedback so the users can verify the computer’s actions, and error correction mechanisms so users can correct the problem and the computer can improve its performance. Most of today’s speech systems require the user to read a transcript of what the computer thinks was said. The only mechanism to correct most “Smart” features or Wizards is to undo the entire operation and try again. AI researchers are trying to make their systems “smarter” and be able to learn faster and more accurately, but it is up to the HCI researchers to work out the feedback and interaction issues that will make these new interaction techniques understandable, effective, predictable, and learnable.