How to Facilitate Effective Computer Use?
The Interaction of Task Complexity and Tool Sophistication

Suresh K. Bhavnani
School of Information, University of Michigan
bhavnani@umich.edu

A synergistic relationship has existed between users, who perform increasingly complex tasks on computers, and system developers who provide increasingly sophisticated tools and interfaces. This synergy has played an important role in the development of extensive and powerful computer systems such as modern computer-aided-drafting (CAD) systems and search engines like Google.

However, despite the above impressive software developments, numerous studies show that many users are ineffective and inefficient in performing computer tasks (e.g. Bhavnani and John, 2000). This observation typically generates heated arguments from different stakeholders on how to help many users achieve their goal of completing tasks efficiently and effectively. **Developers** argue that users need even more sophisticated and intelligent tools, **HCI** professionals argue that users deserve better interfaces, and **educators** argue that users need better training. Who is correct?

This position paper attempts to show that each of the above approaches has strengths and limitations depending on the complexity of the task, and the sophistication of the tools available. Consider a space of *user situations* defined by two interacting variables: *task complexity* and *tool sophistication*. The task complexity continuum ranges from simple tasks (e.g. draw a line requiring a simple mapping to a single command), to more complex tasks (e.g. draw the floor plan of a building requiring the use of many commands). The tool sophistication continuum ranges from simple tools (e.g. such as DRAW LINE) to more sophisticated tools (e.g. wizards that automatically draw shapes based on selections).

At one corner of this task-tool interaction space (simple tasks & simple tools) is the situation where users are faced with a simple task (e.g. draw a line), with simple general-purpose tools (e.g. DRAW LINE). In this situation, the mapping between the two is straightforward. Users essentially have to know where the command resides on the interface, and know how to use it. Here the knowledge to use the command can be largely expressed through good interface design and therefore training in most cases is not required.

However, most users attempt to perform far more complex tasks with simple tools. Moving to another corner of the task-tool interaction space (complex tasks & simple tools) is the situation where users attempt to draw entire floor plans in a CAD system with simple tools such as DRAW LINE, MIRROR, and COPY. In this situation, to be effective and efficient, users need to notice the symmetries in the drawing, and plan out the drawing such that these symmetries are exploited (e.g. draw half of a column and mirror it, then draw half of the bay consisting of many columns and mirror it etc.). Such task decompositions are not easily expressed through current interfaces, and therefore most users require instruction which focuses on (1) how to look for the structure in a task, and (2) how to decompose the task to exploit the structure by using the right commands in the right order. This is where strategies become critical to guide users to notice when, and how to decompose tasks to use effective and efficient commands.

At yet another point in the task-tool interaction continuum (complex tasks & sophisticated tools), users can be aided in the above complex task by more sophisticated tools. For example, the CAD system could provide a *wizard* that asks the user to select options, and automatically creates symmetrical shapes. In such situations, the task of drawing symmetrical shapes is reduced to

selections, but the user *still* has to notice that shapes are symmetrical, and then select and use the correct wizard with correct inputs. Unless there is a tool that completely automates a specific floor plan (leading to a loss of tool generality), there still is a need for training to learn how to understand the structure in a drawing, and the need for good interface design to enable users to effectively use the sophisticated tool.

Finally, good interface design is also important in the situation where users perform simple tasks in the context of sophisticated tools (simple tasks & sophisticated tools). For example, sophisticated tools not only include *innovations* such as the wizard described above, but also sophisticated *variations* of simple tools (e.g. DRAW TANGENT LINE). Tool innovations and variations cause an explosion of tools (also referred to as *featurism*) on the interface, which in turn requires users to acquire the knowledge to make appropriate selections. This complexity can be mitigated through careful menu organizations, default selections, and icon design that are appropriate for users and tasks.

The above analysis suggests that while interface design is important at all points in the task-tool interaction continuum, there are limits to what it can express for more complex tasks. Furthermore, while automation through sophisticated tools can reduce the knowledge to perform certain tasks, they also have limits because the user must know when to use such tools, and because sophisticated tools often trade-off effectiveness for generality. Training appears to be particularly useful when complex tasks are performed with simple tools.

The above space defined by task complexity and tool sophistication is not unique to CAD systems. For example, while Google users can find information for simple facts that occur on many websites (e.g. What is digital zoom in a digital camera?), they have difficulty finding information about more complex topics (e.g. Find a low price for a high-quality digital camera) whose facts are scattered across many websites (Bhavnani, 2003). Using the general-purpose search tools provided by Google, such searches require task decomposition strategies (visit a review site, then a price comparison site, then a discount site) which help users visit different websites in a specific order.

The task-tool interaction continuum therefore helps to reveal the limits of good interface design and the limits of automation in helping users perform complex tasks. In such situations, training becomes important to enable users to become efficient and effective in using complex computer systems.

References

Bhavnani, S.K. Domain-Specific Search Strategies for the Effective Retrieval of Healthcare and Shopping Information. *Proceedings of CHI'02* (2002), 610-611.

Bhavnani, S.K., and John, B.E. The Strategic Use of Complex Computer Systems. *Human-Computer Interaction 15* (2000), 107-137.