The Center for Information-Development Management
710 Kipling Street • Suite 400 • Denver, CO 80215 • 303/232-7586 • www.infomanagementcenter.com

ARTICLE REPRINT

An Application of the Principles of Minimalism to the Design of Human-Computer Interfaces

JoAnn T. Hackos, PhD

This paper was originally presented as a keynote speech at the German HCI conference, Software-Ergonomie '99

Minimalism in information design, specifically as applied to user tutorials and manuals, was introduced in the early 1980s through the work of Dr. John M. Carroll, then a cognitive psychologist at the IBM Watson Research Center. Since that time, theorists and practitioners have further elucidated the principles of minimalism and have attempted to apply it to a variety of situations in which people attempt to learn how to use a software application. Most recently, a new exposition of minimalist principles and practices was published by MIT Press. This work, *Minimalism Beyond the Nurnberg Funnel*, represents the work of leading theorists and practitioners in the field.

I have long been in the habit of describing the user interface as an element of information design and thus amenable to the basic design tenets underlying information design. And, I have frequently characterized the user interface as an expression of minimalist design because the interface combines the traditional design elements of text, graphics, and layout in a two-dimensional space, with the occasional addition of movement, in as compact a form as is practical. More recently, I have been intrigued by the possibility of applying minimalist principles to interface design in a more systematic manner than has been heretofore discussed by its proponents.

In this presentation, I will introduce the four basic principles of minimalism, as well as some of the research done to support the principles, and explain how they might contribute to our understanding of interface design. To the extent which time and space permit, I hope to present some examples of how a minimalist interface might be similar but also differ from more traditional interfaces.

Background to the Minimalist Debate

John Carroll's most developed presentation of the minimalist concept appeared in 1990 in *The Nurnberg Funnel* (MIT Press). In this work, he fully described what he and his team at IBM had learned from observing people trying to learn to use IBM's DisplayWriter word processor and how to learn the computer language, SmallTalk. Learners of DisplayWriter and SmallTalk were most successful when they were provided with brief instructional material that encouraged them to act rather than read. They learned best when the tutorials emphasized the goals they really wanted to achieve, rather than tasks defined by the computer software and the interface.

In one case, for example, the researchers created a cue card that explained to the secretaries how to "Type something" rather than how to "Create a document." "Creating a document" was the heading used on one of the on-screen menus (before Windows) and in the actual DisplayWriter documentation. They had discovered that the secretaries did not want to create documents and were unable to relate their goal, which was to type something, to the name of the task in the interface. This mismatch between goal and execution of the task was a significant impediment to their learning.

In the design of the SmallTalk tutorial, the researchers discovered that programmers were best able to learn the computer language when they were engaged in the actual tasks of writing and debugging code rather than reading about the concepts underlying the language. Users who are encouraged to act, rather than read about acting, are more successful in their learning. Users who are encouraged to perform tasks that are directly related to their goals in using the software are more able to formulate realistic plans, the set of steps they envision will take them through a series of actions to their goals.

Minimalist Principles

In 1996, John Carroll and Hans van der Meij summarized the principles of minimalist design of documentation and training in an article in *Technical Communication*, the journal of the Society for Technical Communication. This article, reprinted in *Minimalism Beyond the Nurnberg Funnel*, most clearly states the four minimalist principles and how they are represented in the design of documentation and training.

The four basic principles of minimalism are

- Principle 1: Choose an action-oriented approach
- Principle 2: Anchor the tool in the task domain
- Principle 3: Support error recognition and recovery
- Principle 4: Support reading to do, study, and locate

My purpose here is to examine to what extent these principles, used to define goals for the development of documentation and training, apply to the design of the human-computer interface.

Principle 1: Choose an action-oriented approach

The first principle of minimalism points to the basic concept that underlies interface design. An interactive interface implies action; users are provided with opportunities for action. In a Windows environment, users select commands from a task bar, make selections in dialog boxes, type information into data-entry fields, and so on. Interfaces ordinarily provide many opportunities for action, even if that action is simply looking at an on-screen report. The intent of basic interface design is to allow users to do something.

However, there is more to the first principle than simply action. An action-oriented approach implies that the user is able to accomplish something. Users are most satisfied, it appears, when they have an immediate opportunity to act and when they believe that the actions they take will lead them toward their goals. Simply clicking on buttons or typing something into a field does not imply that the actions are anything more than random attempts to make progress. In many flawed interfaces, the actions that users take often have little purpose. They are simply attempts to see what, if anything, will result. We observe users trying to guess, unsuccessfully in many cases, which actions will lead to the results they want. The interfaces provide them with little or no guidance about where to start.

I recently reviewed an interface design in which the users are presented with a blank screen and a task bar when they enter the program. Their first cognitive task is to examine the task bar and attempt to guess what item they need to select to accomplish something. Users are immediately confused, unable to take the immediate and purposeful actions that they want to take, because the interface fails to lead them in a clearly defined direction. A more successful design, following minimalist principles, would present an immediate opportunity for action in the context of the users' goals. For example, if the users need to select a patient in order to view that patient's record, the first action available immediately in the interface should be patient selection. Providing an immediate opportunity to act means making the desired activities immediately apparent and reducing the number of choices that the users must make before taking action.

In addition to an immediate opportunity to act, the first principle of minimalism also points to the need for users to explore as a way of learning. Most graphic user interfaces provide more than enough opportunities for exploration, at least in principle. The standard Windows approach is to make many actions available through task bars, pull-down menus, and dialog boxes. Users can browse through the menus, select and review dialog boxes, and make different choices to see how those choices might affect what is happening on the screen. The problem with this approach is that the exploration is undirected. We have found that when users explore at random in an interface, they frequently become increasingly confused, unable to differentiate among explorations that lead to their goals and explorations that lead away from their goals. An interface that guides exploration in productive ways leads the users by using information on the screen that makes clear the direction to take.

Finally, the first principle implies that we must, in creating successful designs, respect the integrity of the users' actions. When we present message boxes that are unrelated to the users' actions, we violate this aspect of the first principle. When we present error messages that fail to inform users how to correct the problem, we also fail to respect what the users are attempting to do. Tips that appear on the screen, unasked and unwanted, distract the users from their tasks. Tips that attempt to track user actions have the promise of maintaining the users' original intent, but, thus far, seem mostly clumsy and overbearing.

One of the researchers at Xerox PARC remarked to me a few years ago that they were having a difficult time making sense of users' seemingly random keystrokes to approach a task. They could not anticipate the users' intent simply from monitoring key strokes. The researcher suggested that a much more sophisticated approach to interpreting the users' logic was needed before such tracking might be successful.

For an interface to comply with the first minimalist principle, to take an action-oriented approach, means the following:

- make critical actions immediately apparent when the users enter the interface, especially as users enter the interface for the first time
- provide opportunities for exploration through the interface, but guide those explorations using appropriate text and graphics so that the connection between actions and goals are easily apparent
- respect the integrity of the users' actions by eliminating annoying and often gratuitous invasions in the middle of tasks the user is attempting to pursue

Principle 2: Anchor the tool in the task domain

The second principle is most significant for the design of user-centered systems. Too often, interfaces reflect the underlying database structure rather than the users' goals. To anchor the tool in the task domain means to design the interface from the users' perspective rather than designing an interface that is functionally correct but disembodied. We have all seen interfaces that fail to resonate with the users' goals. These interfaces have functional names that make no sense to the users. They require sequences of actions that are hidden from view, requiring the users to figure out which task structure will lead them to their goals.

For example, we recently reviewed a Web site designed by a major American university. We took the point of view of a user who wanted to find out if the university offered courses toward a degree in technical communication. We believed this

goal to be typical of many individuals coming to the Web site from outside the university. Unfortunately, the Web site was not designed from the users' perspective, but rather used the organizational structure of the departments and schools in the university to organize the information. Unless the user already knew which school and department offered the courses she wanted to find, she could not get the information she needed. In fact, she needed to know the solution to her problem in order to find the solution—a rather circular approach unlikely to produce success.

In order to anchor the tool in the task domain we must first be well-informed about the goals that the users want to achieve. Do they want to use the task bar or send a fax to a colleague? Do they want to complete the dialog box or find out how many days of vacation they have left? In many interfaces, the users' real goals are obscured, if they were ever understood in the first place. We need to use all of the user-centered design tools to ensure that we truly understand the world from the users' perspective if we hope to build successful interfaces.

The second principle of minimalism exhorts us to build on the users' prior skills, knowledge, and experience in the design of product. As a consequence, we must know the users' goals and relate them to tasks already known in the users' domain to achieve those goals. For example, the software program Quicken uses a metaphorical construct on screen of a check register, similar in appearance to the check register provided by banks with packets of checks. By using this visual metaphor of the task, the designers are able to build upon the users' knowledge of performing the task with the physical artifact. The affordances of the artifact are transferred to the interface. The users are able to bring their prior experience of recording checks and balancing their checkbooks to performing the tasks in the new environment.

Metaphoric constructs alone, however, are not sufficient to reinforce the users' prior skills, knowledge, and experience. In many more complex system designs, the overall relationship among tasks must reflect the users' experience. For example, consider a system to support the task of call tracking, in which the overall flow of information from the primary screen, through secondary screens, and into dialog boxes, is made to resemble the task sequence performed by the users in the physical environment. If we are able to take advantage of known sequences, we spare the users the need to unlearn their previous behaviors and learn new ones. Their cognitive tasks are simplified, the learning curve is reduced, and they are able to accomplish their goals.

In building an interface that respects the second principle of minimalism, we must place information in the interface that supports task performance and enables the users to link the tasks to their goals. To do so often requires more than cryptic terminology. For some reason, we appear to have chosen to preserve the limited language present in the original teletype interfaces. Menu systems often consist of single words when phrases might be more meaningful. Dialog boxes reduce text to field labels and one-word descriptions of radio buttons or check boxes, often making it difficult for the users to know what actions to take.

Given the space we have to work with and the legibility of a graphic user interface, we appear to be constrained by unnecessary brevity, afraid to use words to help the users know what is going on. If we are to support the users' task performance, we should consider adding meaningful language to the screen, using text in phrases, sentences, or even paragraphs. In the past few years, interface designers have introduced the concept of the wizard or coach to assist users in performing certain tasks. Sometimes those tasks have been designed so that novice users are better supported, in effect supplying an alternative to the more cryptic standard interface. In other cases, we have designed wizards to become the primary interface for complex tasks that users have difficulty performing successfully with the cryptic assistance given by the "ordinary" interface. Performance support of these types has proven inordinately successful in improving task performance and enabling users to learn how best to perform tasks and achieve their goals.

It is interesting to note that, in most instances, the performance assistants (wizards and coaches) carry more text explanations than do the "ordinary" interfaces. If we have learned that more text can result in better performance in the context of a well-designed task environment, why not adopt such devices most of the time instead of making them special cases?

In general, on-screen text should provide users with clues about the task structure in the software. Terminology used on the screen should reflect users' goals and lead them through the structure of the tasks. The text on screen, as well as the layout of the information (reflecting reading order or other structures), should help users make the connection between what they want to do and how they accomplish the tasks within the software application.

For an interface to comply with the second minimalist principle, to anchor the tool in the task domain, means the following:

- ensure that the users' goals are well understood by the interface designers so that the interface makes a clear connection between the users' goals and the tasks required to achieve the goals
- build on the users' prior skills, knowledge, and experience by creating a metaphoric structure for the interface that enables the users to connect known abilities to new requirements for interaction
- place information into the interface that supports task performance. Consider using more rather than less text in the interface, along with layout, color, and white space to assist the users in learning how to perform the tasks within the tool.

Principle 3: Support error recognition and recovery
The third principle of minimalism emphasizes the users' need to detect the problems when they occur, diagnose what has happened, and find effective ways to overcome those problems. At the heart of this principle is the need to prevent mistakes before they occur. Often, preventing mistakes requires careful use of default values for fields and choice buttons and error trapping. For example, I was reviewing an interface that required the users to type the hyphens in their U.S. Social Security number (the standard format is xxx-xx-xxxx). If the users failed to include the hyphen, a message appeared stating that the hyphens must be typed. In addition, all the numbers the user had already typed were deleted from the field and replaced with a blank space. Such an approach is insulting to the users and reflects sloppy programming by the developer. First, there is no need to require the hyphens at all. If they need to be added, they can be programmed to appear automatically as the users type. Second, if a particular sequence of characters is required in a field for some reason, and the users make an obvious mistake, the cursor should be returned to the position to correct the mistake following the appearance of the error message.

In many cases, as we have all seen, error messages are both insulting and belligerent toward users. The developers, in effect, accuse the users of being careless and stupid. The developers, in many instances, have chosen to blame the users rather than do the additional programming to prevent the error in the first place.

I am frequently annoyed when I enter a dialog box and have to establish the keyboard focus by clicking the cursor in the first field. Some additional programming is required to place the cursor in the first field, but someone considered that additional programming a waste of time. A decision to waste the time of hundreds, if not thousands of users, to save minutes of programming time should not occur.

In areas where errors are likely to occur or when error correction proves difficult, we need to design in cautions and warnings to alert the users about the potential problems and how to recover. If we provide error messages in such cases, the error messages must be carefully worded to assist the users in detecting the error, diagnosing what happened, and taking corrective action. Too often, error messages simply state the nature of the error, often in rather cryptic language, rather than providing assistance for correcting the error.

One way to help users avoid errors is to provide many default values and explanations within the interface about what might be expected if new values are selected. Warning messages that point to the possibility of future problems are also useful additions, especially when they provide sufficient information to allow the users to make choices about their course of action.

For an interface to comply with the third minimalist principle, to support error recognition and recovery means the following:

- assist users in preventing the error in the first place by using error trapping techniques as thoroughly as possible
- assist users in making the right decisions by providing default values and complete information about the choices available in the interface itself
- create informative, helpful, and courteous error messages that enable users to detect, diagnose, and correct problems as soon as they occur.

Principle 4: Support reading to do, study, and locate
The fourth principle of minimalism, to support reading to do, study, and locate, applies, primarily, to the help system and documentation that you design to support the interface. We know that most of the time users' want to read to do (Redish 1988), seeking information that helps them complete the tasks in the software application and reach their goals. Help systems and documentation that emphasize reading to do, providing task-oriented instructional text, are most likely to assist users in navigating through the application interface.

Users who focus on reading to do profit from context-sensitive help and embedded help systems. Context-sensitive help is designed to anticipate the information the users may need at a particular point in the interface. If the help designer guesses correctly, the help that appears at the users' initial request will solve the users' problem and halt the search for information. Of course, guessing what help is desired is a difficult process and often requires that help developers observe user problems during usability testing of the interface. In some cases, we have recommended developing help text directly in response to information queries during testing. The users ask for assistance; the help developer responds with the minimal information necessary to move the users along; that text becomes the source of the help system.

Embedded help, a system more closely related to wizards and coaches, provides performance support in the context of the application. Since the help is always present, the users do not have to locate the information they need to complete a task. The information is provided to them immediately in the context of task performance. Development tools are now available to embed help within the application screens and to highlight the appropriate step in a process as the users proceed through the activity. (For more information, see the HelpXtender from Wextech Software.)

Well-designed help systems and user guides should be as brief as possible to encourage the users to engage with the software application and apply their previous knowledge and experience to the task. However, it is especially important that the text in the interface, the information on the screen, and the information in the user guides be closely linked to the information in the instructional material. This approach to interface design recognizes that the help system and the user guides are integral parts of the interface and must be designed in association with the interface design.

For an interface to comply with the fourth minimalist principle, to support reading to do, study, and locate, means the following:

- provide minimalist help and instructional text in user guides to support the users' task performance
- develop context-sensitive and embedded help systems so that the transition between actions in the interface and supporting information is as effortless for the users as possible
- carefully integrate the development of all aspects of the interface, including the development of help and additional instructional text so that all elements work together to support the users.

Summary

Minimalism provides us with insight into the design of user-centered interfaces. It de-emphasizes the construction of pleasing screen layouts, placing emphasis more properly on user goal-oriented design. Minimalism suggests, in some ways, that interfaces need to be designed to provide more information to users than they do today. That information might come in the form of metaphoric interface structures, assistance in moving users from their goals through the specific tasks that support the goals. It might come in the form of a richer textual content in screen design and certainly with a focus on selecting the right words and phrases to ensure that users make intelligent choices.

Minimalism also points to the need for close integration among the graphic interface, the help systems, and other forms of instruction. Close integration has been proven successful in the development of performance assistants such as wizards and coaches, which provide more text and more task direction than the typical menu-based design.

Finally, minimalism directs us to be aware of the real goals of our users and to explicitly support the achievement of those goals. Minimalism suggests that we anchor the tool in the task domain, rather than in the database. Such a perspective means that we must spend time understanding the users rather than designing screens.

Bibliography

Carroll, J. M. 1990. *The Nurnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill*. Cambridge, MA: MIT Press.

Carroll, J. M., ed. 1998. *Minimalism Beyond the Nurnberg Funnel*. Cambridge, MA: MIT Press.

Lewis, C. and D. A. Norman. 1986. "Designing for error." In *User Centered System Design: New Perspectives on Human-Computer Interaction*, ed. D. A. Norman and S. W. Draper, pp. 411–432. Hillsdale, NJ: Erlbaum.

Redish, J. C. 1988. "Reading to learn to do." *Technical Writing Teacher* 15:223–233.

BioSketch

JoAnn Hackos is president of Comtech Services, Inc. and Director of The Center for Information-Development Management, which she founded in 1998. She is a partner in SingleSource Associates. She and the Center Associates engage in advising managers of documentation and training organizations in many areas, including single sourcing, developing documentation databases, structuring information, implementing minimalism, and designing and developing quality information for customers.

She lectures worldwide and conducts workshops and seminars that assist communicators and managers in improving their performance and that of their organizations. She has published many professional articles and monographs, contributed to collections and special issues and is known for three significant books, *Managing your Documentation Projects* (Wiley, 1994), *Standards for Online Communication* (Wiley, 1997 with D. Stevens), and *User and Task Analysis for Interface Design* (Wiley, 1998 with J. Redish). She served as STC President in the early 1990s and as editor of *Common Ground*, the journal of the Usability Professionals Association.

Reprinted with permission.

Hackos, JoAnn T. 1999. "An Application of the Principles of Minimalism to the Design of Human-Computer Interfaces." *Common Ground* 9:17–22.