The Need for a Design Lexicon: Examining Minimalist, Performance-centered, and User-centered Design

COLLEEN MACKENZIE

If you have a professional background in technical communication, human factors, usability, computer-human interaction, human performance technology, or instructional design, you most likely believe that good designs encompass knowledge of users, their goals, the ways that they work, and the ways that they learn. The communication products we design apply these values and support users in achieving their goals, efficiently performing tasks, and learning. These are not new concepts, yet the way we talk about them is often ambiguous, especially across different disciplines.

As user support becomes more closely integrated into systems, technical communicators are increasingly becoming integral contributors to the design of user interfaces. We must understand the various design strategies that contribute to effective interface design. Most of the disciplines named earlier have one or two key design strategies. Although these design strategies are unique in their details and represent the disciplines from which they emerged, they all have critical characteristics in common, the most salient being the human element at the center. As a result, the different strategies have overlapped significantly in practice.

Because of this significant overlap, practitioners often use hybrid design strategies. These hybrid strategies may be similar, but their sources and core values are not. Neither is the terminology, even though the terms may mean essentially the same thing. The result is unnecessary confusion and unidentified gaps in beliefs, often to no practical purpose. Therefore, before we jump into the design of performance support systems, we must first distinguish among some key design strategies, their applications, and their goals. At the least, this approach will facilitate better communication across multi-functional design teams. At the most, it could result in more depth in design practice, and we technical communicators can contribute to this result by proposing a cross-disciplinary design lexicon.

This article explores these fundamental design strategies. Specifically, it explores three key design strategies that underlie the development of electronic performance support systems: minimalism, performance-centered design, and user-centered design. Next, this article compares the three strategies and closes with observations on how the three strategies are converging.

ABOUT THE STRATEGIES

Strategy 1: Minimalism

Minimalism has gained wide interest among technical communicators. John Carroll proposed minimalism in the late 1980s after observing novices trying to learn to use computers. He observed that users wanted to use software immediately, but traditional training and documentation slowed them down by emphasizing a “carefully sequenced thorough practice on methods” (Carroll, 1998). In response, Carroll proposed that documentation should observe mini-
Allow learners to start immediately on meaningful tasks.

Minimize the amount of reading and other passive forms of training by allowing users to fill in the gaps themselves.

Include error recognition and recovery activities in the instruction.

Make all learning activities self-contained and independent of sequence.

Figure 1. Primary tenets of minimalism.

Minimalist principles, providing users with the minimum information they need to start real tasks immediately, while allowing them to learn on their own as needed (Carroll 1998). Figure 1 summarizes the main principles that Carroll proposed.

Learning is more effective when it occurs within the context of work (Raybould 1998–2000), and because people want to learn by doing, they need to do real tasks, using their own reasoning skills to enable understanding. In practice, this approach has resulted in the design of modular, action-oriented instruction. Minimalist documentation allows users to start using software in any location and supports them in performing tasks according to the way individual users want to work. As a result, however, users tend to move across learning sequences as they see fit, sometimes causing them to draw incorrect inferences (Carroll 1998); this tendency explains why users need support.

In Minimalism beyond the Nurnberg funnel, Carroll admits that he used to be fixated on the idea of responding to user problems by presenting less information, though he now focuses more on supporting “self-initiated sense making.” This change capitalizes on users’ prior knowledge and lets users transfer knowledge of tasks (in similar or dissimilar domains) to new tasks (Carroll and van der Meij 1998). This method encourages exploration and facilitates learning and retention.

In practice, minimalism has been advocated by the technical communication community and has provided a framework for designing libraries of documentation. Although first proposed for printed documentation, minimalist design has also been applied to online help systems, many of which resemble electronic performance support systems. One particular characteristic of minimalist online help is the application of Carroll’s technique of fading, in which user support is gradually decreased. First, users see the complete set of information. In subsequent viewing, the content is decreased, leaving only cues to the information (Carroll and van der Meij 1998).

Minimalist help systems either reduce the background and non-critical information, or structure information by layering. Layering provides various paths through which users can access information, accommodating different types of users by allowing for selective reading (Farkas 1998). Many commercial software products make use of minimalist design in their help systems by layering information, providing only the most immediately useful information up front, such as task-based instruction. This procedural information is linked to conceptual and reference information and may also include animated demonstrations where the system takes control and demonstrates how to complete tasks. Many of these help systems fail at minimalist design because they include lengthy introductory paragraphs to topics and fail to visually differentiate procedural information from other information that is not critical to performing tasks.

Online help systems that contain wizards and coaches further assist users and minimize the time to performance. Wizards automate tasks so users can perform them quickly and with little or no learning of the application or its user interface. Wizards are especially useful for new users and infrequent tasks. Many user interfaces include wizards, especially in installation programs. Microsoft Works, for example, provides users with the option to create documents through a wizard or on their own. Wizards have minimalist qualities in that users can begin actual tasks immediately. However, wizards violate minimalist design in that they prohibit user exploration and problem solving (Farkas 1998).

Like wizards, coaches assist users while they work through each step of a task. Unlike wizards, however, coaches enable users to interact with the system interface. Coaches let users experiment with the application, advising them when they make errors, suggesting more efficient ways to perform the task, and correcting errors. Although coaches allow users to perform actual tasks immediately, coaches are not entirely minimalist because they force users to perform tasks in a specific order (Farkas 1998).

Strategy 2: Performance-centered design

Performance-centered design (PCD) has gained wide interest among instructional designers and performance technologists. Gloria Gery first proposed PCD in 1995, synthesizing the qualities of effective electronic performance support systems (EPSS) (Raybould 1998–2000). The goal of PCD is to increase overall productivity within organizations by enabling workers to complete meaningful work as quickly as possible (DeLoach 2001). PCD focuses on the performance of humans (users) rather than that of systems, and “infuses tools with knowledge, structures tasks, and enables performers to achieve the required level of performance as quickly as possible . . . with minimum support
1. **Intrinsic support** is embedded in the interface, where it is so closely integrated that users do not distinguish it as separate from the user interface; users merely perceive that they are performing work with the software. One example of intrinsic support is a wizard that steps users through a task incrementally.

2. **Extrinsic support** is integrated with the system but not directly within users' workflow. Online help is an example of extrinsic support.

3. **External support** is not integrated with the system at all. Examples of external support include training, user manuals, and online references (Gery 1995; Marion 1997b).

Although its roots are in training, instructional design, documentation, and human performance technology (HPT, the study of how individuals and organizations can most efficiently and effectively achieve the objectives set out for them), the basic principles of PCD are closely aligned with those of minimalist design. Both reflect the contributions of the usability and human-computer interaction (HCI) fields, in that they show a concern for making computer systems easy to use and investigating how computers affect users (Marion 1997a).

PCD and minimalism differ in that minimalist design focuses on learning while PCD focuses on the performance of work. In addition, minimalism is generic enough to be applied to printed documentation and online help systems. In contrast, PCD is primarily intended for the design of performance support systems. Organizations implement the approach in a continuum that includes embedded (or intrinsic), extrinsic, and external support (Raybould 2000). Raybould suggests that PCD is the implementation of these strategies at the embedded end of the continuum, where support becomes more powerful. As support strategies move toward the external end of the continuum, they become less effective and more expensive (Raybould 2000).

### Strategy 3: User-centered design

User-centered design (UCD) has gained wide interest among human-factors and usability engineers, as well as among technical communicators. UCD was first proposed by the fields of programming and technical communication, and was championed by the usability community. The goal of UCD is to create easy-to-use products (Shneiderman 1998). UCD advocates involving actual users in every phase of the design process, from establishing requirements through iterative testing and post-release benchmark assessments. These users become part of the multi-disciplinary design team, ensuring that the design remains focused on the users (IBM n.d.). Figure 3 summarizes the basic principles of user-centered design.

UCD is similar to minimalist design in that it incorporates user feedback throughout the development cycle.
Shneiderman’s discussion of Carroll’s work on minimalism further suggests that the two design strategies are closely linked (Shneiderman 1998).

Staying focused on users is also a goal of performance-centered design. User-centered design addresses the design process, such as task analysis, competitive analysis, prototype development, and testing, whereas PCD addresses specific design strategies, such as building knowledge and support into the user interface, using wizards, coaches, and other embedded support. The two design strategies are very similar though they are not identical. PCD and UCD differ in that PCD “applies both to the work and to the computer-human interaction alone” (Raybould 2000). Perhaps the best way to consider the relationship is that PCD dictates specific design strategies to employ in UCD; this makes them a complementary pair.

SO WHAT’S THE DIFFERENCE?

What distinguishes these design strategies from each other? Do we even need to distinguish among them? We certainly do. To more effectively communicate across disciplines, we must understand the differences among design strategies. This may be more of an academic need than one needed in practice because actual designs often employ hybrid strategies. But designers from different backgrounds use different terms to describe the same or similar things. By taking the time to understand these strategies and to talk knowledgeably about the differences, we can form an amalgamated vocabulary or cross-disciplinary design lexicon.

Consider the confusion surrounding even the basic definitions of these three design strategies. For example, six separate definitions of PCD are offered as the first item on the frequently asked question list of a resource Web site about electronic performance support (“EPSS terminology” n.d.). The problem is even better demonstrated in a contrast of the definitions of UCD and PCD. In defining performance-centered design, the distinction between the two is that “user-centered design is associated with making software easier to use, whereas performance-centered design is associated with supporting the performance of work” (“EPSS terminology” n.d.). Is PCD, then, not aimed at making software easier to use?

The answer is no. Saul Carliner asserts that PCD focuses on the business goal. That is, the product may be usable, but it may not be useful (Carliner 2001). The following comment was made in a recent Boston globe article on software robots: “Try to imagine any subject, however bizarre, and they will come up with something” (Jesdanun 2001). This seems to exemplify Carliner’s description; we could create agents to do almost any task, but would those agents be useful? Is there a need for them? Do they provide value?

Doesn’t UCD also focus on the business goal? In explaining the difference between PCD and UCD, Winslow and Bramer state, “We don’t want workers to be thought of as system ‘users’ . . . the focus should be performance-centered. We want to enable workers to do their jobs, not focus on having to use a system” (quoted in Marion 1997b). I suggest that this is also a goal of UCD, and the only disparity is in terminology.

Intuit’s TurboTax product is an often-cited example of best practice in the design of performance support systems with its wizards, embedded help, tips, and highly structured reference information. For example, if users do not understand who qualifies as a dependent, they can select the Guide Me option and will be prompted with a series of questions. From the answers, the system determines whether an individual qualifies as a dependent. An excellent understanding of users and their goals is implicit in the design of this system because users are provided with enough information to immediately perform a task while enabling them to learn other areas in the manner in which they want. Is this not also UCD and minimalist design?

In a review of Donald Norman’s Things that make us smart, Dickelman applies Norman’s principles to EPSS and proposes the diagram reproduced in Figure 4 (Dickelman 1995).

This diagram seems to be the crux of all three design strategies. Optimal performance occurs at the intersection of well-designed representations or “cognitive artifacts” that are appropriate to the task and to the person, and that contain critical features of the world (Dickelman 1995). I propose that, along with performance being at the intersection of these three representations, users are at the center of this model, just as users are at the center of all three design strategies.

Each strategy or discipline contributes valuable experience, knowledge, and practices. Depending on the problem to be solved and its context, some strategies may be more suitable than others to produce the best possible
design. The recurring overlaps in these strategies make it difficult to determine the differences between them. The salient commonality is that each discipline endeavors to model systems on characteristics of humans, including the way they work, the environment in which they perform the work, and their goals.

PERHAPS A CONVERGENCE?
Minimalism was one precursor to PCD, which is a subset of UCD. Each discipline promotes its respective design strategy as if it were unique, though they are all intertwined. The goals of each are essentially the same, but they describe design with different degrees of granularity. The distinctions are so slight that they sometimes appear to be only a preference for one discipline's terminology over another.

Other theories and strategies are also related, such as Constantine and Lockwood's usage-centered design (Constantine and Lockwood 1999) and the theory of distributed cognition (Hollan and others 2000). Usage-centered design focuses on how tools are used, not solely on users (Constantine and Lockwood 1999). Is usage-centered design not the same as PCD? From the field of cognitive psychology, distributed cognition proposes that cognitive processes do not necessarily occur individually, but may be distributed across groups or through time (earlier events may affect later events), and may involve interaction with the environment (Hollan and others 2000). This theory of distributed cognition is also related to minimalist design, PCD, and UCD.

Villachica and Stone (1999) affirm that there is no common vocabulary for the performance support system (PSS) approach. This fact makes it dangerous for those who contribute to PSS because their varying backgrounds can lead to “experiential ‘silos’ with widespread lack of communication, continuous reinvention of the wheel, and absence of the progress that could otherwise be achieved” (Villachica and Stone 1999).

We face the same risk with the lack of common terminology across design strategies. This problem is also felt in the usability community, as evidenced by a recent article in the ACM's Interactions: “... [usability's] contributors come from diverse backgrounds and greatly need a shared terminology where meanings are controlled more rigorously than in general vocabulary” (Allen and Buie 2002).

The lack of synergy between the disciplines and these three design strategies may reduce our credibility and others’ confidence in the work that we do in the fields of technical communication, human factors, usability, computer-human interaction, human performance technology, and instructional design. Furthermore, because the design disciplines are so similar, we should merge these design strategies and begin to talk about them together to build credibility as a larger community and to promote understanding of good design practice across specialties.

All three seem to propose that good design

- Lets users perform work efficiently
- Supports users within the natural workflow
- Incorporates business processes
- Is easily accessible
- Is adaptive according to users needs
- Is "easy-to-use" (according to both user perception and defined usability criteria)
- Is what users want
- Provides a pleasing experience

More significantly, although the strategies are not necessarily merging in the academy, the reality is that design strategies are converging in practice. Just as many fields in science merged to form new areas of study, we are experiencing a similar trend in the design community. What we call our design strategy is less important than learning from and applying the principles of each discipline to affect good design.

We must stay informed in all contributing fields and begin to talk about design strategies consistently. A technical communicator may call it minimalist design, the trainer or instructional designer may call it performance-centered design, and the usability expert may call it user-centered design. As technical communicators, we should drive the effort to develop and promote a common lexicon. Even though we have not developed common terminology, we have embraced a common goal: to provide intuitive products that support users in efficiently achieving their goals. We also share the belief that we can all help users efficiently achieve their goals through just-in-time and just-enough information, while letting users learn what they want, when they want.
Finally, we share the belief that at the center of all good design is the user.

REFERENCES


DeLoach, Scott. 2001. E-mail communication.


COLLEEN MACKENZIE is a technical writer for deNovis, Inc., based in Lexington, MA, where she is designing the online support for a Web-based healthcare administration product. Her background is in technical communication and user interface design for software, hardware, and Web-based products. She holds a BA in English and an MS in human factors in information design. Contact information: colleen_mackenzie@yahoo.com