# Minimalism as a Framework

—MICHAEL OATEY
AND MALCOLM CAWOOD

***Abstract*—*We identify various minimalist techniques and argue that, although these techniques can conflict with each other, together they provide a framework for designing computer documentation. A minimalist approach involves making tradeoffs within this framework rather than following a set of prescriptive techniques. Minimalism, in this sense, is a pragmatic design philosophy aimed at the overall objective of "minimizing" obstacles to use. The framework covers the following design issues: word and page count, duplication, selective documentation of facilities, elaboration, task orientation, guided exploration, error recovery, and access.***

***Index Terms*—*Documentation, manuals, minimalism.***

Manuscript received February 1997; revised July 1997.
This paper is based on a workshop on minimalism given by M. Oatey at the Annual Conference of the Institute of Scientific and Technical Communicators, Loughborough University, United Kingdom, October 1993.
M. Oatey is with South Bank University, London SE1 0AA, United Kingdom.
M. Cawood is with Select Software Tools plc, London W1N 9LE, United Kingdom.
IEEE PII: S 0361-1434(97)08969-8.

A RECENT article by Carroll (the founder of minimalism) and van der Meij [1] refers to the "expansive and sometimes confusing literature" on minimalism. They discuss ten misconceptions about minimalism, hoping "that this effort will cause more, not less debate, research, and practical development of minimalism" (p. 72).

This article is a contribution to that debate (although it was originally drafted before the Carroll and van der Meij article appeared). We argue that minimalism is not a prescriptive blueprint that authors can follow mechanically (a point made forcibly by Carroll). We see "a minimalist approach" as a framework of various methods and techniques that are often in conflict but are applicable to different situations. Brockmann [2] talks of a minimalist design "philosophy" (p. 93), but we feel he does not adequately articulate the conflicts in this philosophy. However, Brockmann does point to a rigorous, user-centered approach to designing documentation, which this article elaborates.

The idea that minimalism is a methodological framework finds parallels in the world of systems analysis and design. Glass [3] cites DeMarco's distinction between a Methodology (with a capital M) and a methodology (small m). The former is a process where "all meaningful decisions are made by the Methodology builders" and is seen as "an attempt to centralize thinking." The latter is "a basic approach one takes in getting a job done" and involves different projects needing different approaches (p. 15). We view minimalism as a methodological (small m) framework. Authors need a framework that is flexible and relevant to a multiple objectives.

The term minimalism is used in different ways in the literature. Carroll [4] uses the term in the sense of "minimizing the obtrusiveness to the learner of training material" (p. 7) and of minimizing obstacles to learning (p. 77). He gives particular emphasis to supporting learner exploration and inference, and to *engaging* the user. This may be regarded as the broader meaning of minimalism.

A narrower meaning is minimizing bulk. Of course, the two meanings are related. By reducing sheer bulk, you can reduce obstacles. And by reducing obstacles, you are likely to reduce bulk. But Carroll [4]

stresses: "brevity in itself is more an effect of the minimalist approach than its technique per se" (p. 301).

This article reviews eight minimalist issues and notes how techniques can conflict with each other. The first four issues focus on the narrower idea of reducing bulk; the remaining issues focus on broader ideas of minimizing obstacles to use. The eight issues themselves have been identified from the way the term "minimalism" is used in the literature. The sequence moves from the most obvious issues (word count and duplication) to the least obvious issues (error recovery and access):

1. Word and page count
2. Duplication
3. Selective documentation of facilities
4. Elaboration
5. Task orientation
6. Guided exploration
7. Error recovery
8. Access.

Note that Carroll's work has been in the context of tutorial rather than reference documentation. However, the approach has been extended to reference documentation, see [4, p. 292] and [5]–[8]. We consider both tutorial and reference contexts, and refer to obstacles to learning (tutorial context) and obstacles to use (reference context).

## Word and Page Count

Reducing word and page count is the most common-sense view of minimalism. The aim is to present the same material using fewer words or pages. An obvious means is to reduce verbosity. For an example, see the first expression list at the bottom of this page:

Avoiding verbosity reduces bulk, and it reduces obstacles to use by being clearer.

Using condensed and abbreviated forms also reduces bulk. Dictionaries are an excellent example. The following show similar condensed forms for computer documentation (see the second expression list at the bottom of this page).

However, using condensed forms illustrates how simply reducing bulk does not in itself achieve the minimalist goal of reducing obstacles to use. For example, such condensed forms are particularly useful for quick reference and summary material [8], but such material presumes a competent user whose main need is a readily available answer to a specific problem. Typical examples are an expert user using a half-forgotten procedure, or a transfer user learning the new syntax of a new piece of software. Novice users, on the other hand, may not have the prior knowledge to make much sense of such conventions.

Another obvious means of reducing page count is to reduce white space. White space is "active" when it adds value to a user; for example, as an access aid that facilitates scanning the information. Too much white space (such as margins that take up a third of the page) becomes "passive," increasing bulk and providing no added value to the user. Thus deciding how to use white space is a functional issue rather than a merely aesthetic one. For example, in the dictionary-like example cited below, white space is largely redundant. The main access strategy is through the alphabetic organization. However, if the document is organized according to, say, frequency of use, then white space is more appropriate because this helps users scan the material.

With online documentation, physical bulk is not an issue. However, with only a small display area, navigation is an issue. Horton [9], for example, says:

> Although sparse screens are easier to scan, there are problems... [They] may purchase a 10 per cent increase in reading speed with a 300 per cent increase in the number of screens.... Once users find the topic that answers their question, they read in detail. They do not want to have to plod through three or four screens to read what could be presented on one (p. 236).

**Expression List One**

| | |
|---|---|
| in the event that | if |
| due to the fact that | because |
| in order to be able to | to |
| undertake an investigation into | investigate |

**Expression List Two**

| | |
|---|---|
| from the File menu, choose Save | File>Save |
| hold down the SHIFT key and press Q | SH+Q |

Such fragmentation, especially online, is a serious obstacle. Scrolling through long topics can also be distracting. So the author must choose between a potentially cluttered screen and fragmented text.

A similar problem can occur with the indiscriminate use of graphics. The adage "a picture is worth a thousand words" is a clear minimalist justification for pictures and diagrams. But it invites the retort "Yes, but *which* thousand?" A well-designed graphic that supports the content is justified; a poorly designed graphic inserted mainly to "break up the text" increases page count and can be distracting.

A particular example in computer documentation is screen dumps. Routine displays of screens can double the size of a document without adding any value to the user. Rather, they result in fragmentation, particularly when used with procedures, which can create obstacles to use:

> They [screens dumps] spread apart the steps of a procedure. A half-page procedure now spans four pages. The user cannot easily see how all the steps fit together [10, p. 146].

Screens are often used as checkpoints: to confirm that the user has arrived at the correct place. A possible alternative is simply to give the title of the screen. This is particularly appropriate for online help, especially in context-sensitive help, where the actual software screens are visible at the same time as the help topic.

Of course, some screens are justified: for tutorial material, for complex procedures, and for anxious novices [10]. The argument here is against routine reproduction of most or all screens. Different circumstances demand different techniques.

## Duplication

Duplication is surprisingly common in documentation. For example, Carroll refers to a document where the printing procedure is repeated in four separate variations [4, p. 148]. Apart from increasing bulk, duplication can also create obstacles. One problem is that the user does not always realize duplication is occurring. For example, if different wording is used for the same definition or procedure, the user has to compare the two before realizing there is no difference.

Another problem is that although the document may use the same wording, the repetition simply becomes tedious and begins to "get in the way" of the user. For example, some manuals repeatedly tell the user to enter the customer name in the "Name" field of a form. And how useful is it to repeat in every procedure a basic task, such as selecting text? You not only add three or four steps each time—and thus massively increase bulk—you are likely to irritate the users by forcing them into following the steps for a task they have long since mastered. Price and Korman [11, pp. 238–239] give an example of unnecessary duplication of a four-step procedure (for selecting an object on a screen).

In such cases, it makes sense to "store once and use many times." A basic task like selecting an object or opening a file can be a discrete procedure or topic that users are referred to. Similarly, if a document explains a computer concept or software object every time it occurs, the document would become unusable. Here, glossaries are an effective solution.

However, the issue of duplication is complicated by whether an author wants to create a modular or sequential document, and by whether this format is intended for people learning the software or merely *referring* to a document to solve a particular problem. For example, if someone is working systematically through a document, duplication can be profoundly irritating; clearly signaled cross-references are more effective. However, too many cross-references can increase what Weiss calls "document overhead" [12, p. 16], which obstructs use. As Weiss explains, if the effort needed to *find* the information begins to exceed the effort needed to use the information, then the chances of user rejection increase.

Consider Horton's [9] scenario of a user who simply wants a straightforward answer from an online help system:

> many online documents are used by users who are tired or frustrated, are in the middle of a difficult task, are trying to solve an unexpected problem, and know nothing more about online documentation than how to press the Help key. If accessing information requires disrupting the user's thought process, he or she is likely to continue without the needed information (p. 198).

Such a user is likely to be equally irritated if the online topic they locate does not answer their problem and they are directed to another topic. It is fashionable to say that hypertext jumps overcome this problem, but it is not quite so simple. Too many jumps can quickly lead to disorientation. Hypertext cross-references are particularly disruptive if there is more than one to choose from and if they are lumped, without context, in a monolithic *See Also* section at the bottom of the topic.

On reducing obstacles to use, there are two issues to consider here:

- Should online help systems use self-standing modules of information with hypertext providing links to *additional* information; or
- Should online help systems use hypertext to stitch together re-

> lated information that is stored in different parts of the system?

The latter reduces duplication but, arguably, increases the obstacles and navigational overhead faced by a user.

Again, an apparently straightforward minimalist issue is problematic depending on whether a user (and this can be the same user in different circumstances) wants to use direct or sequential access. There is also the issue of constraints on the supplier. For example, duplicated information might sometimes benefit a user, but requires more resources to maintain.

## Selective Documentation of Facilities

Another obvious means of reducing bulk is to reduce the number of the product's facilities that are documented. This is not as crude as it seems given observations such as: "In almost any program, 80% of the work is done with 20% of the tools or features" [11, p. 176]. Indeed, a short article on minimalism in a British newspaper concentrated almost entirely on the issue of documenting only the "very small subset" of commonly used facilities of a product [13].

Reducing the number of facilities documented also suggests cut-down documentation aimed at specific populations. But not every author is in a position to write custom-tailored documentation to highly specific audiences. A fully comprehensive document is "almost mandated by political, legal, organizational, and managerial concerns, especially for large systems" [14, p. 19]. Indeed, one of the authors of this article, working for a large global company, found that this assumption was unquestionable: the company had an obligation to document everything in the system, perhaps out of simple integrity or perhaps out of a fear of future litigation if errors were to result from omissions in the documentation. Comprehensive documentation is often seen to "complete" the product. So perhaps minimalist-inspired documents must be *additional* to the "official" comprehensive documentation.

For example, Coney and Chatfield [15] compare Microsoft's official *Word 6 User Guide* and the third-party document *Word 6 for Dummies*, describing the latter's greater appeal and accessibility. One feature of the latter document is its concentration on the most commonly used features. However, this is not to say it is a better document; rather it serves a particular audience, whereas the Microsoft authors have different concerns.

The supplier may of course provide different documentation for different audiences. This raises the issue of minimalism for the supplier and minimalism for the user. A cut-down version for a specific population (say, the expert user) is minimalist for that user, but it increases total bulk for the supplier (and demands more resources). Also, if a supplier provides all variations for all populations, each user is faced with deciding what document is appropriate.

Online documentation potentially offers solutions. Horton [9, p. 357], for example, advocates documenting every software object in a product—every menu item, every dialog, every input field—but documenting only the common procedures. By effective hypertext design and layering of the information, users can get started fast, be directed through common procedures, can explore the system, and, if they get stuck, refer for help on any aspect of the system. The downside to this is that too much layering once again increases the navigational overhead.

One elegant solution could be interface annotations such as Microsoft's Tooltips and Apple's Balloon help (pop-up displays that provide brief explanations for each field, menu, or button). Such features mean that every software object is documented while issues of bulk and navigation become irrelevant. Farkas [16] wrote a favorable article about balloon help: he found that this kind of minimal interface-based documentation positively encourages more exploration of the software. This allows other documentation to concentrate on a reduced set of information. Authors must think of the role of the whole interface and user support system and not just of a manual in isolation.

## Elaboration

Elaboration and its implications for bulk are illustrated in this description by Charney *et al.* [17] of two manuals for an operating system:

> One version of the manual, the Elaborated version, contained many definitions, analogies, examples, overviews, explanations, and other elaborations. The Unelaborated version of the manual was about one-third as long (3500 words as opposed to 11 000), and omitted the elaborations (p. 50).

Of course, it is a matter of judgment on what to omit. Some analogies and explanations may go, but definitions of key words are often essential. Carroll himself is particularly critical of overviews, and complains of "layers and layers of introduction, overview, preview, and review" [4, p. 80]. Such elaboration can obstruct Carroll's aim to get users started fast on real tasks.

The concept of elaboration is often characterized as being the opposite of minimalism. Charney *et al.* [17] observe that:

> writers and researchers of computer documentation tend to fall into two quite distinct camps: the "expounders," who believe the instruction should be as complete and explicit

> as possible, and the "minimalists," who believe that instruction should above all be brief and that it should leave much to the learner's own exploration (p. 50).

However, a minimalist author must still make decisions on when to elaborate. For example, the minimalist concept of guided exploration, which we discuss below, is not just about cutting out procedures (and, as is sometimes suggested, leaving the user to thrash around in the dark), but about providing hints, checkpoints, and error-recovery strategies. This implies some sort of elaboration.

The delivery medium also plays a part. For example, take online help. In many respects, it has become received wisdom to create brief "chunks" of online information due to the physical constraints of space. However, brief chunks can cause fragmentation and navigational problems we have already referred to. Thus Horton [9] suggests that:

> If users are jumping from topic to topic, they may find the going rocky. A brief introduction can smooth out the jumps by providing a transition into the content of the topic rather than jumping right in (p. 117).

It is a minimalist goal to reduce elaboration as far as possible. However, this position should not be adopted as a mechanical routine: in some circumstances, some form of elaboration such as introductions can be appropriate.

Consider also the needs of transfer users. They bring to system $Z$ their knowledge of system $Y$. This may be helpful but, equally, it may produce obstacles. For example, the "action grammar" of the Windows-based Excel spreadsheet is "select the required cells then choose a command or action from the menu." On the other hand, the DOS-based SuperCalc spreadsheet is the opposite: the user selects a command from the menu and then types in the range of cells. So to get transfer users started fast, it might be appropriate to provide conceptual material about how the two systems differ rather than force users to work it out for themselves—for example, in the form of overviews, hints, or error recovery.

## Task Orientation

"The central principle in minimalism is task orientation" [1, p. 72]. Task orientation has become commonplace in books on computer documentation. But what exactly does it mean and what is its relevance to minimalism? A task-oriented approach usually involves one or more of the following:

- removing superfluous product information
- organizing around tasks
- using step-by-step procedures
- identifying with the user situation and needs.

By reducing superfluous product information, you reduce bulk. Waite [18] states that a task-oriented document:

> never includes information that is unrelated to the task. A person learning to drive a car may be told how to shift a gear, but not the theory of internal combustion engine (p. WE-38).

A related point is organizing around tasks, rather than around, say, commands. Carroll quotes a survey of 30 examples of quick reference material, which found that all were organized on the basis of system terms rather than user tasks [19]. As Carroll notes, organizing around tasks reduces obstacles to learning but does not necessarily reduce bulk [4, p. 301].

Task orientation is commonly equated with step-by-step procedures. However, Carroll appears to eschew procedural steps: "The problem is not that people cannot follow simple steps; it is that they do not." [4, p. 74]. Organizing around tasks does not necessarily mean using procedural steps. Rather, the author must identify tasks by user analysis, and then seek techniques for supporting those tasks. These techniques include: quick reference materials; examples and scenarios; screen displays without procedures; and guided exploration (see the following section).

The point here is that task orientation, seen as a principle rather than a technique of using procedures, involves finding common ground with the user and so motivating that user. Thus Carroll hopes to "engage the learners by presenting more personally meaningful challenges" [4, p. 147], giving them real, work-based tasks rather than abstract exercises. Price and Korman [11] make a similar point when they say that as technical authors, they seek to "empower people by speaking the familiar language of their work and their action" (p. x in the Preface). Task orientation is thus a way of *identifying* with users.

There are some interesting parallels with rhetoric (see, for example, [15], [20]–[23]). Aristotle's art of rhetoric and its modern manifestations in legal advocacy, advertising, and political propaganda are about *persuasion*. That is not the purpose of technical writing, although Sanders [23] has argued that technical writing does have a "persuasive structure." Rather, we suggest that rhetoric is about outcome: about getting an audience to *act* in a certain way. This is the fundamental purpose of technical communication, and a crucial mechanism is to identify with the audience. As Kenneth Burke [24] says:

> You persuade a man only insofar as you can talk his language by speech, gesture, tonality, order, image, attitude, idea, *identifying* your ways with his (p. 55, original emphasis).

This comment is not unlike one of the main themes of Carroll's *The Nurnberg Funnel* [4]: that the documentation must engage users with meaningful tasks as a basis for motivating them to actually use the software.

## Guided Exploration

Guided exploration is a central tenet of Carroll's minimalist documentation. It is a technique for learning and is Carroll's basis for reducing or avoiding procedural steps. It is also closely related to Carroll's concept of task orientation. In his initial experiments, users were given real tasks, which functioned as motivational levers for the subsequent exploration and more effective learning [4].

Guided exploration aims to support users' natural inclinations to "spontaneously interact" with the system by drawing on their prior knowledge.

> A minimal manual starts with the premise that users always know something, so the manual should not explain what users already know or can easily deduce from the application [5, p. 7].

There is a direct connection with the desire for brevity. A document that emphasizes exploration often eliminates conceptual overviews and uses incomplete procedures. The learning takes place through discovery and inference.

Carroll and van der Meij [1] explicitly reject that exploration means trial and error. As Oatey [25] states, guided exploration lies on a continuum from highly structured step-by-step instructions at one pole to unsupported trial and error learning at the other pole. There can be *degrees* of guidance or support depending on the user and the situation.

Carroll's guided exploration comprises goals, hints, checkpoints, incomplete procedures, and error-recovery strategies. The amount of guidance is the result of user analysis and iterative design and testing. As Carroll and van der Meij [1] say, making a procedure incomplete is not a random process. Rather, it must be grounded in a theory of user inference; it is about teasing out an audience's prior knowledge and skills (pp. 73 and 74).

Although there is much truth in this position for learning, Williams and Farkas [27, p. 42] point out that often a user has no desire for "learning" but simply wants to use the software (for example, people do not learn the telephone directory; they refer to it when needed). Users may well find procedures dull in a tutorial situation, but in some situations (such as Horton's tired and frustrated user) concise procedures are arguably more appropriate.

The whole issue of procedures is problematic. Essentially, we suggest that some procedural information is probably necessary for any documentation that has a reference function, while levels of exploration are most appropriate to training materials. Even in training, procedures can have a role. A minimalist document does not necessarily abandon them altogether; they may be incomplete or truncated or, as Carroll and van der Meij illustrate [1, p. 75], a document may include complete procedures, but with an "on your own" section that invites users to explore for themselves related topics. When developing minimalist tutorials, Vanderlinden [28] decided some procedures were needed for complex tasks, so they used a combination of guided exploration and procedures. One format for this might be a combination of interface annotations and a limited set of procedures (see the end of the prior section on Selective Documentation of Facilities).

The number of steps in a procedure, and the amount of elaboration at each step, is an issue. Users may be prepared to follow five short steps (perhaps leading to further exploration), but not 25-step procedures or page after page of actions in a structured tutorial spelling out of every last action (including every Return key action or mouse click). Vanderlinden's minimalist approach avoided such "handholding" and "painstakingly detailed procedural steps" [28, p. 196].

User personality and learning style also play a part. Farkas and Williams identify "users who cannot master software through trial and error or who have no taste for such a process [29, p. 187]. And Horn suggests that exploration appeals more to "holists" than "serials" [26, p. 8]. Finally, Mirel *et al.* refer to "users who prefer active, exploratory learning-while-doing rather than hand-holding instructions" [30, p. 75]. By contrast, Weiss [12] provides a simplistic, one-dimensional image of human behavior:

> Notice the analogy between documents and computer programs. Manuals affect readers the way programs affect computer hardware.... Manuals or screens pass instructions and data to their readers, who then operate the system correctly and productively (p. 10).

A minimalist approach rejects such a one-dimensional view.

Of course, the situation as well as personality influences user behavior. Carroll [1] found that:

> even learners who declare that they are the type of person who reads everything before trying to do anything, quickly and spontaneously begin interacting with the system as they read (p. 76).

But those same people may well be bewildered when, say, they consult an online help system to find a quick answer and are invited to explore the system. The likely result is a rapid rejection of the documentation

as useless—the ultimate obstacle to any learning or use.

However, authors of reference material can still learn something fruitful here, namely, the emphasis on inference and induction. Williams and Farkas [27] rightly point out that this can be dangerous. There is often no independent validation of whether that inference is correct or the best practice. But in certain circumstances and for particular users, inferring a solution may be the best or indeed the only realistic way of providing instruction.

We are thinking here specifically of scenarios and examples, which illustrate a solution to a complex procedure that might have hundreds of variables. For example, with Paradox for Windows relational database software, instructing users how to create queries or forms is almost impossible using a step-by-step format. No procedures can realistically cover every button click, while any chosen sequence of actions is purely arbitrary. Particular queries or form designs are effectively infinite, and relate to both the structure and purpose of the database that users have designed. Thus it seems that a better technique is to provide users with several fully developed examples that use as many of the variables and options as possible, so that users can infer how best to use the facilities for their own purposes. Indeed, recent research by IBM [31, p. 368] found that expert and experienced users actively craved this kind of documentation.

Learning by example is a valid technique in any type of computer documentation. It also reduces bulk. The choice on whether to use it depends, again, on analyzing the users, their objectives, and the nature of the software.

Perhaps the biggest practical problem with guided exploration is the emphasis on constant testing and modification—documentation as permanent revolution. This is undoubtedly a desirable state of affairs, but how many authors have the time or resources? Authors wanting to start from a minimalist framework must recognize that their options may be limited before they even start. Simply cutting procedures in a random or haphazard way is to be rejected out of hand. As Williams and Farkas [27] note:

> We certainly endorse careful adaptation to different audiences and reiterative developmental testing of any documentation, but at the same time we note that it is far safer and easier to strive for completeness and clarity (p. 43).

This is not simply taking the easy option, but a recognition of external constraints that affect what choices authors choose—or are able—to make. In this position, some minimalist techniques are more realizable than others.

Nevertheless, authors should at least consider alternatives to comprehensive and elaborated procedures. The alternatives include:

- using condensed or truncated procedures (as in some quick reference cards);
- documenting interface objects comprehensively (as in balloon help);
- providing examples and scenarios;
- supporting guided exploration.

These techniques allow for user inference. They may be additional to comprehensive procedures, or they may reduce the size and number of procedures.

## Error Recovery

Some writers particularly emphasize the error-recovery aspect of minimalism (for example, [32]). Error-recovery techniques certainly go hand-in-hand with exploration. If users do not follow structured information and procedures, they need more support for recovering from the errors that inevitably occur. And because effective learning can follow from errors (and their diagnosis and solution), the aim is to avoid the consequences of errors rather than the errors themselves [25].

However, if the documentation is primarily for reference and is centered on procedures rather than exploration, is there still a role for error recovery? The ideal answer is, surely, "Yes," but there are attendant issues and tradeoffs. To be effective, troubleshooting sections still require close analysis of the audience (to assess likely user problems) and of the software (to assess any weak links, or "hotspots") [1, p. 74]. Access to this information must also be made easy. All this adds bulk and, with complex and customizable software, can result in potentially massive documentation.

An alternative strategy can be to provide telephone support (see, for example, [6, p. 11]). With complex software, this may be the only realistic solution as it becomes impossible to preempt every possible error. Similarly, online error messages provide error recovery (although this depends on how understandable they are; esoteric system messages are useless and can actually increase anxiety and reduce motivation [33, p. 305]). As we discussed at the end of the section on Selective Documentation of Facilities, what is included in the manual or online help system has to be offset against what is included in the interface itself.

## Access

The least obvious of the minimalist issues is access, but it is implied in Carroll's suggestion of allowing reading in any order: "materials designed to be read in any order cannot be read in the wrong order" [4, p. 84]. This requires rapid direct access.

Techniques that allow direct access are modular organizations and al-

phabetic sequence [34]. Mirel *et al.* [30] discuss two-page modules for active learners, and suggest "the layout makes information more accessible to users who explore on their own and refer to text as needed" (p. 84).

Indexes allow direct access, though Carroll [4] surprisingly suggests users prefer browsing to using an index (p. 84). The following quotation shows an interesting tradeoff:

> Because minimalists consider an index superfluous, it was eliminated, and the tasks were arranged alphabetically to facilitate searching [5, p. 8].

When designing minimal manuals, Draper and Oatley [32] propose an "information flow" approach and design around user action; users do not so much remember information as remember how to access information as and when required for action. This again implies rapid access mechanisms.

Access is more concerned with reducing obstacles than reducing bulk. Indeed, techniques such as indexes can increase bulk, as can increasing white space to assist scanning access. On the other hand, tradeoffs with duplication can decrease bulk. And bulk itself is one of the biggest obstacles to rapid access. The cycle of tradeoffs continues.

## Conclusion

Writers on minimalism have emphasized different issues and techniques. Essentially, there are two dominant themes—the commonsense exhortation to reduce bulk and "be brief," and the broader ideas of supporting user exploration and inference. For the technical author, these themes might easily be dismissed: to write briefly is no more than a truism, while issues like guided exploration seemingly have no relevance to reference documentation.

However, taking all the issues together as an overall framework, we believe that authors can enrich their design. This involves tradeoffs, with inevitable conflicts and dilemmas. Williams and Farkas [27] use the word "dilemma" pejoratively; but we suggest it is a source of creative tension, a lever for effective design.

However, issues are not always in conflict. For example, one design shows the application of three minimalist issues: selective documentation of facilities, elaboration, and guided exploration. This is a "staged" manual, which starts with a minimal core of commands, then moves on to an expanded set of commands, and then the remaining commands are

> presented in slimmed down version, perhaps only by a quick reference sheet, or guided exploration hints, or even an "ultraminimal" manual which simply lists the tasks to attempt (relying on the system to deliver any information needed) [32, p. 239].

An underlying theme of this paper is identifying with users: engaging them so they use the documentation. Reducing bulk clearly does this by making the documentation more approachable and less formidable, and by easing physical access and use. Minimalism also engages users by focusing on real tasks and by supporting modes of working such as guided exploration.

The result is not necessarily a super slim, brief, or truncated document. Rather, we cannot say in advance what a minimalist-influenced document will look like. Depending on the users' needs and the purposes and context of the documentation (for example, tutorial or reference, comprehensive or third-party, paper or online), it might or might not include: graphics, duplication, cross-references, glossaries, guided exploration lessons, developed examples and scenarios, procedures (incomplete or otherwise), overviews, troubleshooting sections, and indexes.

## Acknowledgment

The authors wish to acknowledge the contribution made to an earlier draft of this paper by Anna Pollard of South Bank University. Thanks are also due to Malcolm Beaumont, Peter Critten, and Alan Pool for their comments.

## References

[1] J. M. Carroll and H. van der Meij, "Ten misconceptions about minimalism," *IEEE Trans. Prof. Commun.*, vol. 39, no. 2, pp. 72–86, 1996.
[2] R. J. Brockmann, *Writing Better Computer Documentation: From Paper to Hypertext*, version 2. New York: Wiley, 1990.
[3] R. L. Glass, "Through a glass darkly," *Data Base for Adv. Inform. Syst.*, vol. 27, no. 1, pp. 14–16, 1996.
[4] J. M. Carroll, *The Nurnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill*. Cambridge, MA: MIT Press, 1990.

[5] J. Scholtz and M. Hanson, "Usability testing a minimal manual for the Intel SatisFAXtion faxmodem," *IEEE Trans. Prof. Commun.*, vol. 36, no. 1, pp. 7–11, 1993.
[6] J. S. Craig, "Using a structured design analysis to simplify complex in-house computer manuals," *IEEE Trans. Prof. Commun.*, vol. 35, no. 1, pp. 7–12, 1992.
[7] J. E. Ramsey and K. Oatley, "Designing minimal computer manuals from scratch," *Instruct. Sci.*, vol. 21, pp. 85–99, 1992.
[8] M Oatey, "Minimalism and quick reference documentation," in *Colloquium on Issues in Computer Support for Documentation and Manuals*. London, U.K.: Inst. Elec. Eng., Dig. no. 1993/169, 1993, pp. 2/1–2/3.
[9] W. Horton, *Designing and Writing Online Documentation: Hypermedia for Self-Supporting Products*, 2nd ed. New York: Wiley, 1994.
[10] ____, "Dump the dumb screen dumps," *Tech. Commun.*, vol. 40, no. 1, pp. 146–148, 1993.
[11] J. Price and H. Korman, *How to Communicate Technical Communication: A Handbook of Software and Hardware Documentation*. Redwood City, CA: Benjamin/Cummings, 1993.
[12] E. H. Weiss, *How to Write Usable Documentation*, 2nd ed. Phoenix, AZ: Oryx, 1991.
[13] D. Ince, "At last, a few words of advice," *The Independent*, Oct. 14, 1991, p. 18.
[14] C. Hallgren, "The Nurnberg Funnel: A minimal collection," *J. Comput. Document.*, vol. 16, no. 1, pp. 11–17, 1992.
[15] M. B. Coney and C. S. Chatfield, "Rethinking the author-reader relationship in computer documentation," *J. Comput. Document.*, vol. 20, no. 2, pp. 23–29, 1996.
[16] D. K. Farkas, "The role of balloon help," *J. Comput. Document.*, vol. 17, no. 2, pp. 3–19, 1993.
[17] D. Charney, L. Reder, and G. Wells, "Studies of elaboration in instructional texts," in *Effective Documentation: What we Have Learned from Research*, S. Doheny-Farina, Ed. Cambridge, MA: MIT Press, 1988, pp. 47–72.
[18] R. G. Waite, "Organizing computer manuals on the basis of user tasks," in *Proc. 31st Int. Technical Communication Conf.*, Soc. Tech. Comm., 1984, pp. WE 38–40.
[19] P. Reitman, "Streamlining your documentation using quick references," *IEEE Trans. Prof. Commun.*, vol. 31, no. 2, pp. 75–83, 1988.
[20] C. E. Beck, "Systems theory and rhetoric: A theoretical model of communication," *Tech. Commun.*, vol. 42, no. 1, pp. 133–141, 1995.
[21] P. Brooksbank, "Rhetorical considerations for the humanization of online help," in *Proc. IEEE Int. Professional Communication Conf.* (Banff, Alta., Canada, Sept. 28–Oct. 1, 1994), pp. 361–365.
[22] M. J. Killingsworth, "Toward a rhetoric of technological action," in *Proc. 37th Int. Technical Communication Conf.*, Soc. Tech. Comm, 1990, pp. CC 17–19.
[23] S. P. Sanders, "Building better bridges: Persuasive structure in technical writing," in *Proc. 37th Int. Technical Communication Conf.*, Soc. Tech. Comm., 1990, pp. CC 23–25.
[24] K. Burke, *A Rhetoric of Motives*. Berkeley, CA: Univ. Calif. Press, 1950.
[25] M. Oatey, "Trial and error versus instructions," *Communicator*, vol. 4, no. 7, pp. 7–8, 1994.
[26] R. E. Horn, "Commentary on the Nurnberg funnel," *J. Comput. Document.*, vol. 16, no. 1, pp. 3–11, 1992.
[27] T. R. Williams and D. K. Farkas, "Minimalism reconsidered: Should we design documentation for exploratory learning?" *SIGCHI Bull.*, vol. 24, no. 2, pp. 41–50, 1992.
[28] G. Vanderlinden, T. G. Cocklin, and M. McKita, "Testing and developing minimalist tutorials: A case study," in *Proc. 35th Int. Tecnical. Communication Conf.*, Soc. Tech. Commun., 1988, pp. RET 196–199.
[29] D. K. Farkas and T. R. Williams, "John Carroll's The Nurnberg Funnel and minimalist documentation," *IEEE Trans. Prof. Commun.*, vol. 33, no. 4, pp.

182–187, 1990.
[30] B. Mirel, S. Feinberg, and L. Allmendinger, "Designing manuals for active learning styles," *Tech. Commun.*, vol. 37, no. 1, pp. 75–87, 1991.
[31] G. Mitchell, "Shrink-wrapping the formula for better information: End user vs MIS computer documentation preferences," in *Proc. IEEE Int. Professional Communication Conf.* (Banff, Alta., Canada, Sept. 28–Oct. 1, 1994), pp. 361–365.
[32] S. W. Draper and K. Oatley, "Action centred manuals or minimalist instruction? Alternative theories for Carroll's minimal manuals," in *Computer and Writing: State of the Art*, P. Holt and N Williams, Eds. Oxford, U.K.: Intellect Books, 1990, pp. 222–243.
[33] B. Shneidermann, *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 2nd ed. Reading, MA: Addison-Wesley, 1992.
[34] D. K. Farkas, "Encyclopaedic reference as a model for print documentation," *Tech. Commun.*, vol. 36, no. 2, pp. 122–125, 1990.

**Michael Oatey** is a Lecturer in Technical Communication in the School of Computing, South Bank University, London, United Kingdom. He was a technical author in the computer industry for five years, and specializes in self-instructional methods. He received the M.S. degree in education from the University of Wisconsin, Madison, and the Ph.D. degree in media selection from the University of London.

**Malcolm Cawood** is a technical author with Select Software plc. He received the M.Sc. degree in technical communication from South Bank University, London, United Kingdom, and has research interests in rhetoric and online documentation.