QUALITY DOCUMENTATION -- A HOW-TO

by
Robyn
McIntyre

Good documentation begins with a plan. While the plan presented in this paper isn't by any means definitive, it should help you get started and keep you going while you develop a formula of your own. It consists of four main elements:

- Defining the need
- Planning the document
- Writing the document
- Maintaining the document

DEFINING THE NEEDS

Before anything else, you should define the needs the document will be designed to meet. Doing that involves asking three questions:

- What is being documented?
- Who will be reading it?
- Will it also serve as a sales tool?

What is being documented?

What purpose will the document serve? Will it be used as a training manual, a procedures manual, a reference manual?

Who will be reading it?

Will the document be written to serve a technical audience, a non-technical audience, or someone in-between?

Will it also serve as a sales tool?

Almost every document meant for commercial purposes has the capacity to be used as a sales tool. Well-written documentation is an increasingly important factor in the buying and selling of software and hardware products. But some documents can be more sales-oriented than others. A manager's guide, for instance, might be expected to contain more information about the features, benefits, and reliability of a system than a key-entry operator's manual.

When you have defined the needs the document will serve, you will be better able to focus on what it must contain. And that will be a guide to you as you go on to the next stage of creating a document: Planning.

PLANNING

Planning is probably the most important task in the process of developing a document. For many it's also the most time-consuming and boring of the tasks (having a big wedding is often more fun than planning it). But without good groundwork, a document (like a wedding) can turn out to be a disaster. To plan for a good document involves four steps:

- Formatting
- Research
- Writing an Outline
- Estimating Writing Time

Formatting

If your company provides you with a word-processing system, but doesn't provide you with a "style" guide, formatting will be

Planning
FORMATTING

your first step in planning the document: You need to know where headers will appear, what your margins will be. Rough drafts needn't be <u>that</u> rough, and even with a wp system, you can save time by <u>not</u> having to go back and change margins and such. If you're starting from scratch, designing the "look" of your document may involve looking at other manuals, and consulting publications on technical writing. The main idea is to produce a document that presents its material in a way that is both effective and pleasing to the eye. The look is important because if it doesn't appear inviting, no one is willingly going to read it. While you're designing your format, take into consideration the type of media that will be used. For the purposes of this discussion, I'm assuming the final result will be a manual. But obviously your approach will be different if the final "document" will be microfilm, video tape, or plastic-coated cards like the instructions found on copy machines.

Whatever media you use, there are some elements of formatting that provide a "cleaner" appearance and better communication:

- headers
- call-outs
- bullets
- narrow columns
- simple pagination
- visual transitions
- illustrations

headers

Almost everyone will make use of one kind of header or another, but many will not consider the real value of a header as a signpost. A bolded header on every page will help the reader to determine quickly what information is to be found there. Without having to dig into the text. If possible, place a subhead under the page header (see "FORMATTING" at the top of this page) to further define the information being discussed.

call-outs

Call-outs, like the ones found in the margins of this paper, will help your reader zero-in on the specific information he wants.

bullets

Imagine a string of grocery item names in paragraph form... The moment you see it, you begin to mentally separate the items into a list. Don't make your readers do this extra work. List-type information is easier to read and to understand if it is shown in list form, and bullets will give the information continuity and emphasis.

narrow columns

Paragraphs should be short and fairly narrow. Text is easier to read when the information is broken up into small pieces. Without anything to divide it up, the text becomes confusing, the eye tires, and the reader starts to wonder whether it is worth the effort.

simple pagination

The simplest page numbering system is the best. Finding page 1-10 rather than A-202.003 is not only easier on the reader when he's looking for something, but will make maintaining the document an easier task for you.

visual transitions

Visual transitions are another way of breaking the text up into more easily-digestible pieces. They may be borders, or horizontally-ruled lines.

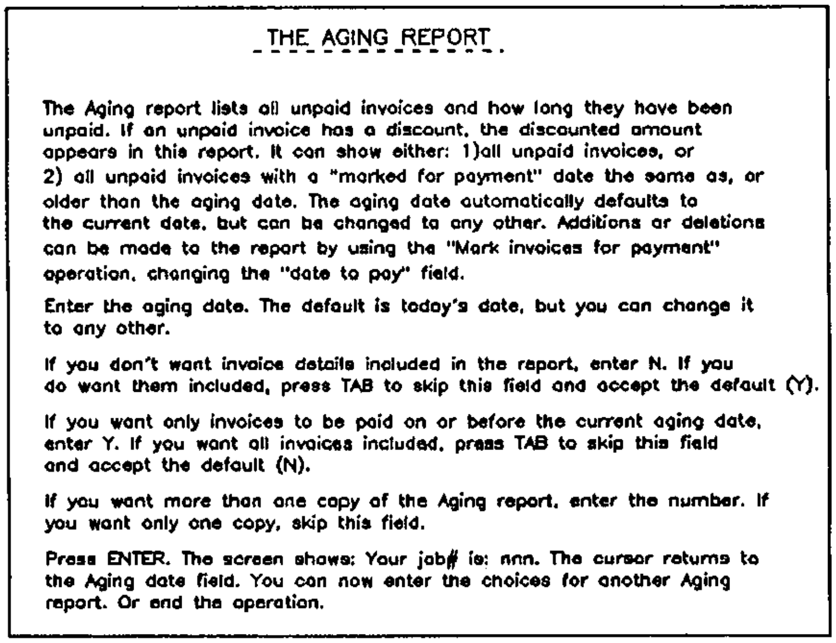
Planning
FORMATTING

illustrations

If your budget or technical resources permit, include as many illustrations as you can. They will make your point faster, and if done well, will aid your reader in more thoroughly comprehending the material. In addition, they make the document more attractive, and serve as visual transitions.

To clarify the points I've made about formatting, let's look at a couple of illustrations.

Figure 1.



THE AGING REPORT

The Aging report lists all unpaid invoices and how long they have been unpaid. If an unpaid invoice has a discount, the discounted amount appears in this report. It can show either: 1)all unpaid invoices, or 2) all unpaid invoices with a "marked for payment" date the same as, or older than the aging date. The aging date automatically defaults to the current date, but can be changed to any other. Additions or deletions can be made to the report by using the "Mark invoices for payment" operation, changing the "date to pay" field.

Enter the aging date. The default is today's date, but you can change it to any other.

If you don't want invoice details included in the report, enter N. If you do want them included, press TAB to skip this field and accept the default (Y).

If you want only invoices to be paid on or before the current aging date, enter Y. If you want all invoices included, press TAB to skip this field and accept the default (N).

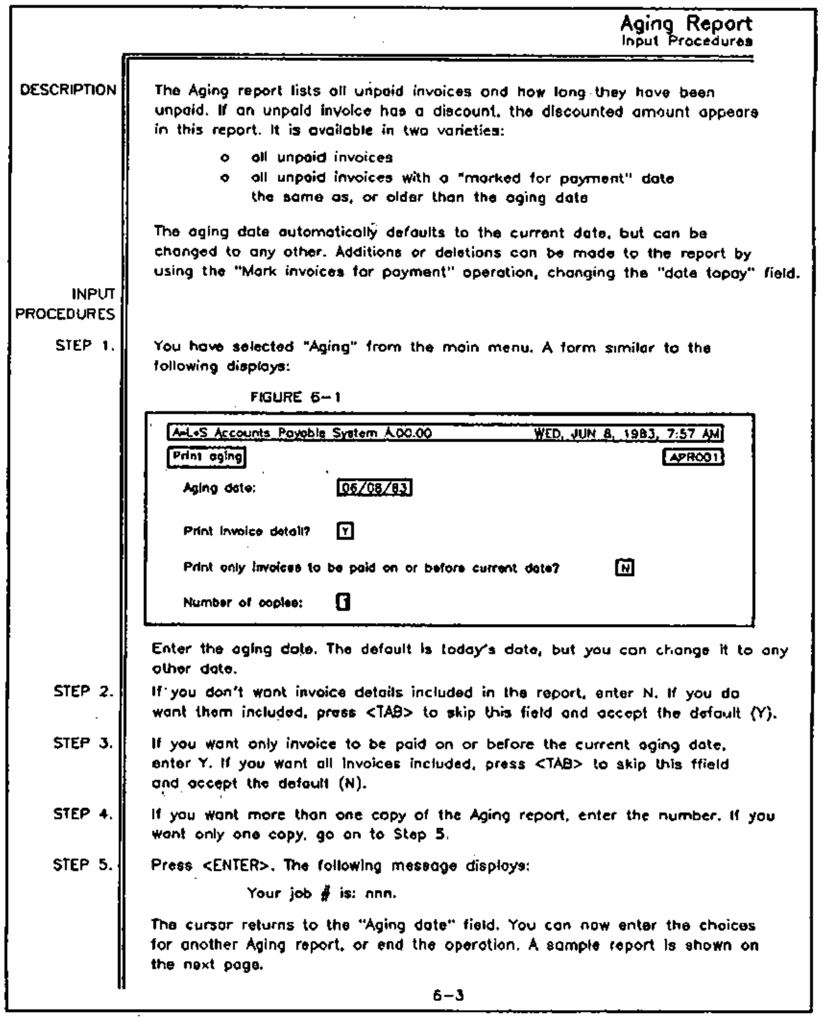
If you want more than one copy of the Aging report, enter the number. If you want only one copy, skip this field.

Press ENTER. The screen shows: Your job# is: nnn. The cursor returns to the Aging date field. You can now enter the choices for another Aging report. Or end the operation.

Compare Figure 1 to Figure 2, an example of better formatting, and the inadequacy of Figure 1's presentation is thrown into sharper focus.

Planning
FORMATTING

Figure 2.



Aging Report
Input Procedures

DESCRIPTION

The Aging report lists all unpaid invoices and how long they have been unpaid. If an unpaid invoice has a discount, the discounted amount appears in this report. It is available in two varieties:

- o all unpaid invoices
- o all unpaid invoices with a "marked for payment" date the same as, or older than the aging date

The aging date automatically defaults to the current date, but can be changed to any other. Additions or deletions can be made to the report by using the "Mark invoices for payment" operation, changing the "date topay" field.

INPUT PROCEDURES

STEP 1. You have selected "Aging" from the main menu. A form similar to the following displays:

FIGURE 6-1

```
A-L-S Accounts Payable System A.00.00            WED, JUN 8, 1983, 7:57 AM
Print aging                                                          APR001
  Aging date:              06/08/83

  Print invoice detail?    Y

  Print only invoices to be paid on or before current date?     N

  Number of copies:        1
```

Enter the aging date. The default is today's date, but you can change it to any other date.

STEP 2. If you don't want invoice details included in the report, enter N. If you do want them included, press <TAB> to skip this field and accept the default (Y).

STEP 3. If you want only invoice to be paid on or before the current aging date, enter Y. If you want all invoices included, press <TAB> to skip this ffield and accept the default (N).

STEP 4. If you want more than one copy of the Aging report, enter the number. If you want only one copy, go on to Step 5.

STEP 5. Press <ENTER>. The following message displays:

Your job # is: nnn.

The cursor returns to the "Aging date" field. You can now enter the choices for another Aging report, or end the operation. A sample report is shown on the next page.

6-3

Though Figure 2 doesn't have bolded headings (it was created using HPDRAW), it still entices the eye easily through the information it presents by the use of call-outs, bulleting, and obvious paragraphs. A likeness of a screen the user might see is included, and an allusion is made to a report sample. Whenever possible, illustrations of what the reader may see on a CRT screen or in a report should be included.

Once you have at least a rough idea of what the finished document will look like (bearing in mind that minor alterations may still be necessary), you can get on to gathering the information to be presented.

Planning
Research

Research

Researching the information you'll be including in the document involves three tasks:

- defining the information to be included
- learning how the system works
- setting up interviews

defining the information to be included

If the system already exists, you'll need to gather any previous documentation written for it. Although documenting after-the-fact goes more quickly than creating documentation for a system still in the design phase, the inconsistencies you are likely to find cannot be readily corrected because the system is in use. If the system is still in the planning phase, make sure that you are included in the creation cycle; that you get a copy of any outlines, flowcharts, et cetera, which illustrate what the system will consist of. Get a copy of the bug list and ask to be kept informed about changes, fixes, and scheduled completion dates (very important, if you're accurately to estimate writing time). If regular meetings between the project team; programmers, testers, documentors, engineers, or others are held, attend them. As boring as a meeting often is, unless it's a disguise for a gripe session, some valuable information about the way the system works will be discussed there.

learning how the system really works

Approach the system from a user's perspective: sit down in front of a terminal and use it. In an ideal evironment, the completed system or the module you're working with has already been tested and released for use. To do a thorough job of documenting, you will still need to learn the system, but if a Test department exists, find out who is testing your software. The tester can often provide more detailed information about how the system works than the programmers can (especially if there are several of them coding the project). Unfortunately, of course, some companies don't have Test departments. In that case, it's crucial that you learn the system. You'll probably find several inconsistencies in menus and form names, and a few bugs that can be corrected at least in the next release. Needless to say, take lots of notes. Indicate where the way the system actually works differs from the way it was expected to work. In particular, try entering erroneous information. You may find an error message doesn't exist where it should, or (as I've seen happen) you may even blow up the system. Let me emphasize that the object of this exercise is not to catch the programmers out, but to help them make sure that the system works the way it's supposed to, with no unpleasant surprises. And don't hesitate to indicate changes you feel might make using the system easier; fewer menus, more menus, changing the name of an operation, and so forth. Remember that you are representing the user, and you should keep their best interests at heart.

setting up interviews

Make a list of the people you should talk to before and during the document's creation. This includes the programmers and testers, and if the system is in use (and you have access to them), the people who use it. Talking to the users is especially important because, after the documentation is printed and released, you will need feedback on its accuracy. We'll discuss that further in Maintaining the Document. For now, make a list of people to talk to, and when you have several questions that need answers, schedule interviews with the appropriate persons.

Planning

Writing an Outline

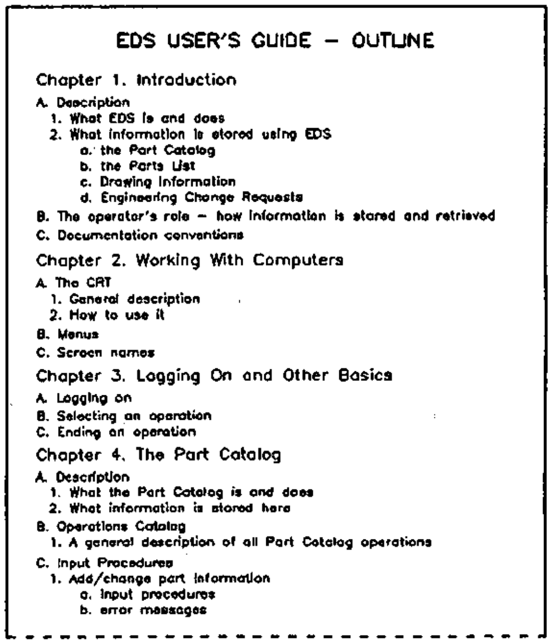
Now that you've determined what the document will look like and what it will contain, it's time to put together an outline.

Writing an outline

Begin by determining the organization of the material: although it may sometimes be advantageous to begin at the main menu and go right down the list, it often makes more sense to find out which operations or modules will probably be used most frequently and therefore should be more prominently featured. For example, if reports aren't scattered hither and yon, it might make sense to group the procedures for printing them in a chapter of their own titled "Reports". Try to have the information flow from "most-used" to "least-used". If the document will be a tutorial, begin with the simple and progress gracefully to the complex. For example, help the user to be successful at creating and printing a basic letter before you discuss intricate formatting commands and techniques for mass-mailings.

Figure 3 is an example of an outline for an engineering system user's guide.

Figure 3.



EDS USER'S GUIDE – OUTLINE

Chapter 1. Introduction

A. Description
   1. What EDS is and does
   2. What information is stored using EDS
      a. the Part Catalog
      b. the Parts List
      c. Drawing Information
      d. Engineering Change Requests

B. The operator's role – how information is stored and retrieved

C. Documentation conventions

Chapter 2. Working With Computers

A. The CRT
   1. General description
   2. How to use it

B. Menus

C. Screen names

Chapter 3. Logging On and Other Basics

A. Logging on

B. Selecting an operation

C. Ending an operation

Chapter 4. The Part Catalog

A. Description
   1. What the Part Catalog is and does
   2. What information is stored here

B. Operations Catalog
   1. A general description of all Part Catalog operations

C. Input Procedures
   1. Add/change part information
      a. input procedures
      b. error messages

Looking at Figure 3, you'll notice that Chapters 2 and 3 consist of basic information needed to use the system. The step-by-step instructions for performing operations begin in Chapter 4, and once the key entry clerk has mastered the basics, Chapters 2 and 3 can be skipped over with no loss of information. Chapter 1, of course, is an introduction to the system. In this, the user's manual, the introduction is less detailed than might be designed for a manager's guide, since the key entry clerk needs less background and interpretive information to do his job than someone who must plan for an entire department.

Planning

Writing an Outline

Your own outline may be more or less detailed, depending on how much information you feel is necessary to remind you about what must be included and where. Try to keep wide left and right margins because you will probably scrawl notes as you progress. And if you use a word processing system to create the outline, leave it on the system. Information may arrive late and make revisions necessary.

When you're satisfied with the outline, have it approved. When I worked for AM Jacquard, approval of an outline was required from the manager of documentation. Yours may be approved by the DP manager, programmer, or no one. The important thing is to have someone familiar with the material review the outline and give you some feedback about how the information will be presented: Did you leave something out? Does the material progress logically? Approval of the outline can be especially important if you are working closely with the programmer and user to design a new system. Your outline may be the first opportunity they have to see, in a rough way, how the software will work. After reviewing it, they may decide some operations have been left out, and others are redundant. Using it, they can decide whether the project is on-target. And you can fine-tune your organization of the information.

Now that you have pretty much decided what the document will look like, begun your research, and have gathered enough information to decide how the document will be organized, it's time to estimate how long it will take to write it.

Estimating Writing Time

If you are documenting software that already exists, and has been released to you for documentation, estimating the time needed to write it will be fairly straightforward. Figure 4 shows a formula you can use (I'm indebted to Progressive Communications of Colorado for developing this specific formula).

Figure 4.

TASK	TIME ESTIMATE
First Draft	Research time + writing time + 15%
Second Draft	25% of time to create 1st draft
Total writing time	1st draft time + 2nd draft time
Total proofing/ editing time	25% of total writing time:
1st draft	75% of total proofing/editing time
2nd draft	25% of total proofing/editing time
Total typing time	35% of total writing time (automated typing is assumed)
1st draft	75% of total typing time
2nd draft	25% of total typing time
Review	Double the reviewer's estimate!

Planning

Estimating Writing Time

Now let's take a look at what this formula looks like when translated into reality. Figure 5 shows the time estimated for completing a first and second draft for a report guide containing information about fifty reports.

Figure 5.

Writing/Production Time
Report Guide – 50 reports

TASK	HOURS
First draft = 1 hr. research + 2 hrs. writing + 0.45 hrs. (extra 15%) =	3.45
Second draft = 25% of 3.45 hrs. =	0.86
Total writing time = first draft (3.5 hrs.) + second draft (0.86 hrs.) =	4.3
Total proofing/editing = 25% of 4.3 hrs. = (1st draft proof/edit = 0.8 hrs.) (2nd draft proof/edit = 0.27 hrs.)	1.07
Total typing time =35% of 4.3 hrs. = (1st draft typing = 1.13 hrs.) (2nd draft typing = 0.38 hrs.)	1.5
Total writing/production time to finished 2nd draft (one report) =	6.9 hrs. per report
Total writing/production time for 50 reports = 6.9 hrs. x 50 (reports) =	344 hours (div. by 40) = 8.6 weeks

As you can see, using this formula, the estimated time for completing the work is over eight and one-half weeks at five days a week.

When estimating time for the first draft, take into consideration the time you'll need to arrange for, or construct roughs of tables, report samples, and examples. Will they be hand-drawn or printer-generated and reconstructed by the art department? Must you include screens in the draft or can you just indicate where they will be included later? If you are your company's art department, reports, tables, examples, and screens will affect how long it will take to complete the final draft. So before you estimate when writing will be completed, make a list of all of the illustrations you'll need and what it will take to get them all together. If other people are involved in the process, take into account that even though you are all working on the same material, you won't work at the same speed.

As handy as the formula is, though, if you are documenting software still in development, you may not be able to use the formula to estimate a finish time for a completed document. The program may not be ready when you are. In cases like this, it's best to work in a module form, breaking down the time necessary for documenting the entire program into its component parts. For example, if I'm writing an instruction manual for an engineering system and I know the reports are finished, though nothing else is, I can give an estimated time for documenting the report section. Documenting software as it's released means you and the programmer will have to work closely; you're depending on him to stick to his schedule so that you can stick to yours. If you have a situation like this, make as sure as you can that you have a means of prodding him if he slacks off. And that the prodding is done through a third person such as the project leader, DP manager, or software manager. This will ensure that the programmer won't duck down a stairway when he sees you in the hall.

I'd like to note at this point that the formula for estimating writing time may become entirely useless to you if you must work backward from a due date. But (at least in my experience) unless the due date is entirely unrealistic, you'll probably have more time than you need to finish up.

Finally, after completing the four steps necessary to planning the document (formatting, research, writing the outline, and estimating writing time), you can get down to writing.

WRITING

Although planning is the most time-consuming and possibly boring task connected with creating a document, most people (including me) will do almost anything to avoid actually beginning to write. I sharpen pencils (seldom needed when using TDP), get a cup of coffee, ask my friend Cathleen about her wedding plans; anything to avoid that moment when I face the terminal and begin the first sentence. I don't know why this is so; I've been writing user manuals for six years, and fiction for longer than that, but I always have this reaction to a new piece of work. So if you have that same feeling, don't think you're alone. Writing is hard work. You are creating something. Maybe it isn't destined to win you the Pulitzer, but it will still have your own stamp on it, and it will take energy (a lot) to produce.

Plan to write poorly

To minimize start-up anxiety, keep a couple of things in mind: Since you're writing from an outline, you can begin wherever you feel most comfortable. Also, you can plan to write poorly. This means, don't be too concerned with the style and sentence structure in the first draft. If you have stockpiled material from various sources, throw it in without a thought for how it sounds and whether the transitions are rough. You can always (should always) go back and clean up the text. The main thing is to get started and get it all out. Still, planning to write poorly doesn't mean you shouldn't do the best you can while you're at it; the better your organization and writing is to start with, the less editing it will require. And if you keep the following elements in mind as you write the draft, you'll be ahead of the game:

- tell him, tell him, tell him
- control jargon
- use stock phrases
- keep it conversational

Writing

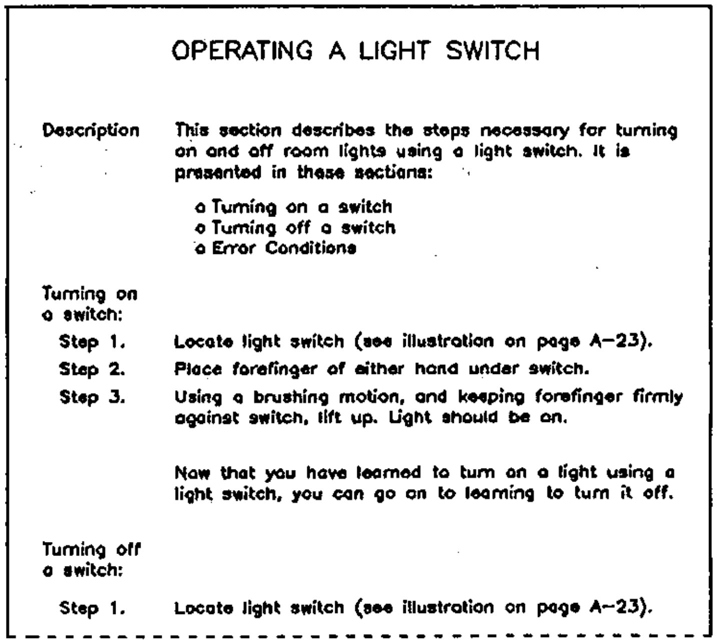
Plan to Write Poorly and Edit Well

- stay active
- keep it simple
- build signposts
- avoid cuteness
- include necessary references
- provide thorough examples

tell him,
tell him,
tell him

This refers to an old saying in military teaching: Tell them what you're going to tell them, tell them, tell them what you've told them. Translated, it means you'll start out by telling the reader what he can expect to find in a chapter, then you'll tell him what you said you would, then you'll tell him what it is he just learned. This redundancy will make it easier for the reader to remember what he's been told, and it will help you move logically from one place to the next. How about a simple example:

Figure 6.



OPERATING A LIGHT SWITCH

Description — This section describes the steps necessary for turning on and off room lights using a light switch. It is presented in these sections:

o Turning on a switch
o Turning off a switch
o Error Conditions

Turning on a switch:

Step 1. Locate light switch (see illustration on page A-23).
Step 2. Place forefinger of either hand under switch.
Step 3. Using a brushing motion, and keeping forefinger firmly against switch, lift up. Light should be on.

Now that you have learned to turn on a light using a light switch, you can go on to learning to turn it off.

Turning off a switch:

Step 1. Locate light switch (see illustration on page A-23).

You may think that repeating yourself is likely to irritate the reader, but it doesn't work out that way. It will actually make it easier for the reader to remember what you've told him. Just remember while you're repeating yourself to avoid talking gibberish.

control jargon

For many readers, jargon is gibberish. They don't have your job or associates, so they probably don't hear the same catch phrases every day. If you insist on using buzzwords to describe processes, they will probably give up in disgust and frustration. Sometimes (as with data processing), some jargon is unavoidable. A new data entry clerk will more than likely have to learn to use the word "CRT" even if he doesn't know its exact definition. When you must use jargon, explain it. Or include a glossary of terms used frequently in the documentation. Try to keep the number of terms small, though; people visiting foreign shores don't usually immerse themselves in the language of the country they're visiting. One thing that will help are stock phrases.

stock phrases

Stock phrases such as, "The operations performed in the ------ feature are:", or "The cursor returns to the ------- field" become familiar to the reader and make him more comfortable with the instructions. They also make writing the document go faster because you aren't creating a new phrase for each function to be performed. Don't worry about creating these stock phrases before you begin your first draft. If you look for them as you write, they will more or less identify themselves to you, and indicate where they belong. But new text or stock phrase, keep it conversational.

keep it conversational

A document explaining a technical process is bound to be dry; don't completely dehydrate it by using a tone more suited to addressing a gathering of stuffed shirts. Keep it friendly, keep it light, keep it conversational. You're explaining something to a friend, face to face. Keep your text on that level.

stay active

One way to keep the tone conversational and interesting is to stay in the active tense. My english handbook defines active voice in writing as "making the subject of a sentence the doer of the action", as in

"Ned washed the car."

When the subject of the sentence is the receiver of the action, then you have passive voice, as in

"The car was washed by Ned."

Just from these two examples it's obvious that the first sentence (active voice) not only uses less words, but is more lively and interesting. If you have been taught to use passive voice, you may find it difficult to catch yourself at it. I recommend that you do two things: Get a reference book on english usage and study up on active and passive voice, and find someone who can spot passive voice when she sees it. Ask her to read through your draft when it's finished and indicate the passive constructions. After some help you'll find it easier to recognize them on your own, and then you'll find them cropping up a lot less often.

keep it simple

Passive construction is one of the chief reasons that many scientific journals and publish-or-perish papers are so darned boring. Another reason is that, for one reason or another, the authors seem to feel their viewpoints won't be taken seriously unless they throw in a lot of jaw-breaking vocabulary. Maybe so, but your reading audience is more likely to be interested in learning how something works, and not in how educated you are. This doesn't mean that you can't use polysyllabics, but it does mean that you must take your reader into account. Go back to where you defined the needs the document is being designed to meet. Who is going to read it? If it's designed for managers, you could use a more complicated vocabulary. But manager or key entry clerk, the issue isn't intelligence but time. Will the reader have the time and patience to deal with a lot of ten-dollar words. The

Writing
Plan to Write Poorly and Edit Well

answer is probably "No". So keep it simple. Don't patronize the reader, but remember that he has to get through an awful lot of technical information in as short a time as possible. Make it easy for him to cut through the text by saying what you've got to say in the plainest possible way.

build signposts

And while you're building plain text, you might be constructing a few signposts. Signposts are transitions; the sentences that lead the reader from one section to another. Try to make the transition an easy one; an abrupt transition will shock and confuse a reader enough to make him lose his concentration. Once concentration is lost, it's difficult to regain. Reading, as well as writing, is an energy-consuming task. Many people wouldn't read at all if they didn't have to, and if you make it tough for them, they certainly won't read <u>your</u> work. Which brings us to another concentration-breaker: cuteness.

avoid cuteness

Cute drawings and sayings can be useful in some circumstances; sales material, mostly. But cute drawings have three failings:

- they break the reader's concentration
- after a few readings, they stop being cute
- the style helps to "date" the material

On the whole, your document will be better off without drawings about "Willie Workorder" or "Penny Purchaser". It will also be better off if you use the space to include necessary references.

include necessary reference

What is a necessary reference? A necessary reference is a bit of information about another command or procedure or piece of hardware that has a bearing on the information you're currently describing. A good example is something that recently happened at my office: The department secretary was formatting a document using TDP. The end of the document contained a command to include another TDP file. It wouldn't format. Each time she tried, the formatting would abort, and it took her, me, and the DP manager 15 minutes to figure out that it was aborting because the file to be included specified an environment file in the first line, confusing TDP. Using hindsight, the problem might seem obvious. But first you have to make the connection. How much simpler to avoid the problem entirely by including a short note in the documentation for the IN command stating that the file to be included cannot contain an environment file specification! Don't make things any harder for the reader than you must. If some command or procedure affects another command or procedure, indicate it.

provide thorough examples

Another way you can make things easier for the reader, is to provide thorough examples. It's not necessary to try to cover every contingency, but the examples you include should provide enough information to enable the reader to use the command or procedure with confidence. Don't give weak or incomplete examples. Incomplete information causes readers to give up, and a command or procedure that isn't used isn't really useful.

Okay, the first draft is finished. You've planned to write poorly, but saved yourself a lot of grief by remembering to:

Writing
Plan to Write Poorly and Edit Well

- tell him, tell him, tell him
- control jargon
- use stock phrases
- keep it conversational
- stay active
- keep it simple
- build signposts
- avoid cuteness
- include necessary references
- and provide thorough examples

Now what? Now, of course, you edit.

Edit well

By editing, I don't mean proofreading. If possible, someone else should check to make sure the commas are in the right places. No, by editing, I mean reading through the work to see if it makes logical sense; does one thing follow another in a sensible manner so that the reader is led gently through the material to its obvious conclusion? From first entering the data to printing the report? From writing a basic letter to creating complex contracts from boilerplate?

To save your sanity, you will probably want to break down the editing process into several readings:

- check for organization
- check for transitions
- check for sentence construction
- check for completeness and content

Unless you used to teach grammar, you could probably use a good reference book. You'll find plenty of them in the reference book section of your local bookstore, but if you can, I recommend you buy Strunk and White's Elements of Style (MacMillan Company, publishers). A good reference book will become invaluable. After a few documents, you'll develop a feel for sentence variety, agreements, and other fun grammatical stuff, and you'll need your reference book less often.

Proofreading

Once the editing has been done on the final draft, the document can go to the art department, or official proofreader, or whomever can do a good job at ferreting out misspellings, punctuation errors, and the like. Try not to do this job yourself, if you can avoid it. After being totally involved with the document, you can't see it the way someone else would. If you must proof it yourself, try to give it some cooling off time. Drop it in a drawer for a couple of weeks. When you do see it again, it will seem different to you.

Miscellaneous

Before we leave editing for maintenance, I'd like to make some miscellaneous points about reviewing documentation:

Preliminary Stamp

Try to have a stamp made up to say something like, "Preliminary Version" or "Rough Draft". Use them when you release the first draft of a document for review. Sometimes rough drafts get copied or lost, and if you're doing in-house documentation, you won't want anyone using the rough draft and thinking it's gospel. Also, stamping it "Preliminary Draft" or something similar will remind reviewers that they are not looking at the final version.

You might not think that would be necessary, but you would be surprised at the number of times I've had to tell someone, "Don't bother about that; the work isn't finished."

Reviewers

Speaking of reviewers, make sure a technical reviewer knows what his job is: reviewing a document for technical accuracy. Many reviewers assume that they should help you out by editing your work and proofing it, too. This can be maddening, since you've already budgeted time for proofreading and editing. And if the reviewer wastes time on these tasks it will only be longer before you can get to work on the technical content revisions. That's assuming that the reviewer knows his editing and proofing stuff. If you've explained this to the reviewer, and gotten nowhere, try Progressive Communication's suggestion: have him indicate technical content changes in one color and editing/proofing changes in another color. That way, you can easily identify the editing/proofing marks, and ignore them.

Indexing

One thing you shouldn't ignore is the index. Any complicated piece of documentation should have an index, and as complete an index as possible. What constitutes a complicated work? Anything written where there is so much information valuable to the reader that you can't list it all in the table of contents, or where an item is discussed in more than one chapter or section.

When in doubt about whether or not to include an item in the index, I recommend including it. It's easy to skip information you don't want, but very irritating to be unable to find what you do want.

MAINTENANCE

Finally, you've dealt with the reviewers and gotten the document formatted, printed, and released. Now you must maintain it.

Bug Lists

If you're part of a large programming/documentation staff - as in a research and development shop - word will come down to you via the bug/enhancements list about what changes you can expect the document to need for a specific release.

If you are a one-person documentation department, there isn't any reason why you shouldn't ask for a bug/enhancement list if one doesn't already exist.

However your company works, you must be informed about expected changes well in advance or you can't plan for writing them up. Don't let your manager get away with coming up and saying, "Oh, by the way, we changed the Blank Report six months ago and forgot to tell you. I need a revised copy of the Report Guide for tomorrow's staff meeting. Get to it, will you?" You'll only get ulcers. I may be predjudiced, but if somebody hires me for a job, I expect them to let me do it. And do it right. To that end, I also expect some management cooperation. So if you're currently working for a company without a method for informing you about programming changes that will affect documentation, come up with one, or work with your boss and the programmers to come up with one, and insist that they use it.

Feedback

Another type of documentation change to be considered is the change indicated by user feedback. Using postage-paid forms, telephone or in-person interviews, it can be possible to find out

Maintenance

what the reader thinks of the way you organized and presented the information in the document. This kind of data is often difficult to get, though, since many readers will find work-arounds for inaccurate documentation without telling the vendor that they're necessary.

Modular Approach

Once you've become informed about expected changes, you've got to make plans for ensuring the changes make it into the documentation. If you're working for a software/hardware vendor, the changes should be included in the appropriate release. And if you've used a modular approach to documentation (Report Guide, Input Procedures, etc.), you should just be able to instruct the reader to remove Section X. from the manual and replace it with the new Section X. This cuts down costs, and makes both your task and the reader's a lot easier.

In-House Library

In-house documentation should be catalogued in a company library. The documentation department, or document control department, or whoever, should have a list of all manuals produced and where in the company they're located. This will speed the process of replacing the old with the new.

Page Breaks

If, when you wrote the document, you avoided putting in hard-coded page breaks except where necessary, the task of adding or subtracting text will be eased. (Of course, by hard-coded page breaks I mean entering "\NEW" commands or anything else that would force a new page.)

Table of Contents

While you're listing the parts of the document to be added to, or subtracted from, don't leave the table of contents out. Changes in the document may mean additions or deletions here, too.

Index

And by all means, don't forget about the index. If you're adding new information, the index may also require additions or deletions in even more detail than the table of contents.

CONCLUSION

Well, we've finally reached the end of this paper, and I still feel I could have been more specific. Unfortunately, there is not time or room for an in-depth study of the art of writing quality documentation. That would be (and has been) a subject for an entire manual, three-day seminar, and even a college quarter. I hope, though, that I have given you a good basis to start from by stating that the formula for writing quality documentation consists of four elements:

- Defining the need
- Planning the document
- Writing the document
- Maintaining the document

As I have said, while planning the document is the more time-consuming task, writing the document is the most difficult. So go easy on yourself if you're new to tech writing; realize that it takes time to learn, and lots of practice to do something well. Don't be afraid to borrow ideas (unless they're copyrighted); each writer has his or her own style and will inevitably create a different product using the same tools. If you've been a technical writer for awhile, and you enjoy it, you've already noted the ideas most interesting to you and no doubt made plans to incorporate

Conclusion

them into your next work. But beginner or pro, if anything I've covered in this paper helps you make your documentation clearer to the reader, I'm satisfied. Because that was always my main objective.

Robyn McIntyre is Senior Technical Writer for Infotek Systems in Anaheim. When not staring blankly at CRT screens or waylaying programmers, she can often be found researching her current project, "The Best Chocolate in L.A. (County)".