DON'T LET YOUR PROGRAMMER GROW UP TO WRITE OPERATIONAL DOCUMENTATION ---

OR SHOULD YOU?

J. B. Watterson
ORI/CALCULON Corporation
P.O. Box 270
Germantown, MD 20874

SOME FOOD FOR THOUGHT

How many of you actually enjoy writing and maintaining documentation? I thought so. At the risk of incurring the reaction "Oh not again!", I submit that you can actually enjoy developing online system operational documentation. Although there are no magic remedies for documentation problems, there are some tools and practices to assist us in creating good, useful documents. Creating good documentation can be easy and it can be fun! The best way to accomplish this is to make documentation development an integral part of the programming and maintenance process.

After all, you are accustomed to working at your terminal. You actually enjoy programming, and aren't afraid of the computer hardware or software. With some guidelines, techniques, and encouragement, you can document. And, you might even enjoy doing it.

Let's first look at some observations.

- Most of us don't like to write! Since we don't enjoy writing, we often don't create the documents we need. And when we do create them, they are often unreadable. Unreadable documents are unusable documents.

- Most of us would agree that an application is only as good as its documentation. Yet documentation is typically inadequate, and is a neglected by-product of the software development process.

- Poor documentation adds costs. User training costs increase because it takes longer to bring the user up to speed. Maintenance costs are higher because program applications are not documented properly - internally or externally.

- Good documentation requires some extra effort on the part of each of us. If provided with guidelines and time, documentation can become a fun and challenging part of your job. You will accept and enjoy your documentation responsibilities. The result -- good documentation.

So, let's take a brief look at how we do it today:

- Our systems documentation generally consists of a System Reference Manual and a System Maintenance Directory.

- Traditionally, operational documents are created as paper documents, maintained by support programmers or technical writers, and distributed by a technical reference librarian.

OUR APPROACH TO OPERATIONAL SYSTEMS DOCUMENTATION GENERALLY CONSISTS OF A SYSTEM REFERENCE MANUAL FOR USERS, AND A SYSTEM MAINTENANCE DIRECTORY FOR SUPPORT PROGRAMMERS.

We have taken the general FIPS PUB Federal government standards and designed customized operational documentation guidelines tailored to our environment. These guidelines provide for consistency without rigidity. Our documentation then reflects the needs of the users of a wide variety of systems.

Our structural approach to operational systems documentation generally consists of:

- System Reference Manual for users.

- System Maintenance Directory for support programmers.

The System Reference (User) Manual describes interactive terminal operating procedures and software/operational capabilities for user organizations. We use non-ADP terminology. This manual is the primary user document, serving both first-time and experienced users.

The manual provides complete instructions for use of the system by the users. It is used as the primary reference during user training. It is the standard reference to which users turn during production operation of the system. This document is structured in such a way that its effective use does not depend on prior data processing experience on the part of the users.

The scope of the manual includes initiation of system operations, data entry and correction, production of reports, and system recovery procedures. A hardcopy manual complements the systems online menus, screens, and prompts.

Quick Reference Cards (QRCs) are also provided, as needed, for quick and easy assistance to the users. They provide easy access to information that users need often while they are online, such as codes or special commands. The QRCs are printed on bright, heavy card stock, in a "Z" fold fashion. The cards are easily maintainable and can be distributed with inexpensive plastic cases which users can attach to their terminals.

The System Maintenance Directory is designed to acquaint the maintenance programmers with the system processes, system programs, and files. It serves primarily as a road map to the entire online system. It identifies the components of the system and their interrelationships. The directory generally includes an introduction, system overview, processes, program names and files. Files include data base schemas and data sets. Supplementary information, as required, is included as an appendix.

Unlike the Program Maintenance Manuals of the past, we no longer include flow charts, program descriptions, or a detailed logic flow for each program. Information of related standard software packages that can be obtained easily from other sources are also not included.

Program-level documentation for each program is part of the source code and is updated whenever a program is modified.

The operational documentation, hardcopy or online, should be easy to use, and simple to maintain. Yet, it must provide the necessary information for the users and support personnel to run and maintain the system.

> TRADITIONALLY OPERATIONS DOCUMENTS ARE CREATED AS PAPER DOCUMENTS, MAINTAINED BY SUPPORT PROGRAMMERS OR TECHNICAL WRITERS, AND DISTRIBUTED BY A TECHNICAL REFERENCE LIBRARIAN.

For years, system user reference and program maintenance documentation have been created as paper documents during the operational documentation phase of the systems development life cycle. The programmer scribbles these documents (reluctantly when they find the time). They are typed, printed and distributed by the hundreds by the technical reference librarian.

Then the poor maintenance programmer has to try and update the documents (again reluctantly) and the cycle continues. If you are lucky enough to have a technical writer, you can be assured the grammar and spelling are correct and the i's are dotted. And don't forget those distribution lists have to be updated -- or are they?

Current documentation development practices are expensive, difficult to manage and maintain, and often lead to products of inconsistent quality. I'll ask my question again -- how many of you actually enjoy writing and maintaining hardcopy documentation? I think you are getting my message.

My history lesson is over. Let's talk about some tools and practices you can use to enjoy developing good documentation.

Here's how we can revolutionize our documentation procedures:

- We need to use our computers as information gathering and distribution tools. Drop the paper and pencils.

- o Write like you talk. Reach the point where you can write in a conversational style. Use words that are easy to understand and avoid jargon. Write as if the reader is:
  - Unprepared.
  - Insecure.
  - Confused.
  - Incompetent.
- o We need a commitment to user-friendly communications: online documentation.

These are the areas we are going to discuss from here on in this presentation. First, this isn't a 1, 2, 3 "how to" presentation, but one to give you some ideas and make you think about them. Take these ideas back to your own organization and implement the ones that will work in your environment.

WE NEED TO USE OUR COMPUTERS AS INFORMATION GATHERING AND DISTRIBUTION TOOLS

For years, we have been providing the users with systems to gather, edit, manipulate, store and report information. Documentation is information. So why haven't we used these system capabilities? One obvious answer -- no shoes for the shoemaker's kids. We need to use our computers as information gathering and distribution tools for operational documentation.

We have our editors, HP Text Data Processor (TDP), word processing, graphics packages, and now desktop publishing. We have the computers, albeit mainframe, HPs and the PCS. And, we have communication packages like HPs Distributed Systems (DS). So let's start using these tools to develop our operational documentation. We still need some hardcopy documentation, but we need lots more online documentation. So start using some of your computer hardware and software to your advantage.

Why?

1. You will love it. This is your natural habitat. You feel warm and cuddly with it, and can develop documentation as you program. If you use a word processor, it probably features a spelling check, perhaps even editing features. Some shops have put together some menu interfaces to create documentation skeletons you can fill out at your terminals. And, there are commercial packages available to assist with documentation. By using some of the methods I'm about to discuss, you can easily write like you talk and create much of your documentation online.

And wouldn't it be easier for you if the inventories of your application programs were stored as online data sets? They would be easier to update than having them printed in a hardcopy System Maintenance Directory.

2. Documentation is easy to maintain when it's online. Whether updated by a programmer, a technical writer, or a word processing person; modifications can be made easier and quicker. Editor bars can be inserted, date of modification indicated and new pages added. Even a revised table of contents can be generated automatically, if needed. Desktop publishing puts the means for producing quality hardcopy operational documentation in the hands of the developers.

3. Most users like online documentation. For one of our new major systems, we developed both online documentation and a paper System Reference Manual (SRM). We found that the paper document was used infrequently. We have about 80 active users on this system, but only a dozen hardcopy SRMs are now available. Our users prefer using the online documentation and the Quick Reference Cards (QRCs). For other systems, we have database managers updating access tables and error message tables. The maintenance programmer no longer has to perform these clerical chores.

For one of our large, complicated budget systems, we developed five QRCs. The first card contains general information, i.e., title, system modules, points of contact, logon/logoff procedures, and command conventions. The other four cards represent each of the system modules. With these QRCs and online documentation, only a few of the users needed a hardcopy System Reference Manual.

4. If we develop documentation using the computer, we can use the computer to distribute technical documents. For example, our Departmental Accounting System, operating on the Hewlett-Packard, provides a method for storing all operational systems documentation as files, and transferring text from/to the host HP to the word processing equipment. The modified HP files are transmitted electronically to the ten accounting sites for internal printing and distribution.

Someday we hope to have online user documentation for this system, but it is always easier to develop it for new systems than to go into an existing system. Documentation experts estimate that it costs from 4 to 10 times as much to document software after it is written than to do it while a program is being developed.

5. Finally, there has to be a cost savings associated in using our computers as information gathering and distributing documentation. The savings are generated by providing online documentation at time of programming and eliminating much of the hardcopy printing and distribution. When these documents are updated online by the maintenance programmer and released to the production system, they are immediately available to the user. A major savings in costs and other resources.

It is virtually impossible to determine the price that we pay for faulty documentation. We may know what it costs to document something properly, but what does it cost to not document? Consider, for example, the costs of training and maintenance for a poorly documented system. It takes users longer to come up to speed, and how much longer does it take a maintenance programmer to fix a bug? Who knows?

I had one programmer develop all of his documentation on the HP3000 using TDP and an HP100 lap-top PC. Then he devised a way to transfer it to an IBM PC, which was connected to our NBI word processing machine. The word processing folks did some reformatting, clean up, and spelling checks. They printed a beautiful draft copy for review and modifications.

Others have done something similar using the IBM mainframe and word processing packages on a PC and transferred the document to our centralized word processor. My point is they enjoyed it. It worked, and the company saved money.

Something that can't be measured in dollars is the online documentation psychology. The programmers and users like it. Management is pleased. Good documentation is available and not a forgotten entity. We were electronically sending documentation for one system, but realized that no one was copying and distributing it. To our embarrassment, our own programmers hadn't even gotten a copy of the revisions. The online documentation psychology makes for high morale, updated information and cost savings. Try it -- you'll like it!

WRITE LIKE YOU TALK

We write it as if we are trying to write. And we shouldn't. We should write it as if we are speaking. We ought to talk it. Because in talking it, we automatically put some emotion into what we are saying. We eliminate the technical, archaic, flowery, meaningless, tongue-twisting wordiness. This comes with getting our pencil or terminal into gear before our brains have thought it all out. We get diarrhea of the fingers.

Now let's discuss some reader-based writing techniques. These will make it easy for you to develop good documentation -- and even enjoy it.

Have you ever done much writing? Probably not. Few of us ever feel an itch to write. The rest - 99% or more of us - were born non-writers. Writing has always been an unpleasant chore - something like a visit to the dentist or a car dealership! Most everyone has to do a certain amount of writing in their job.

Why don't we enjoy writing? The reasons are numerous. But one common reason is we are afraid our grammar, punctuation, usage and spelling are not up to snuff. Therefore, we are afraid to write -- we hate writing. But

you don't need more grammar, punctuation or usage (spelling maybe) to develop good documentation.

Most of us need a basic change in attitude. Let's look at a new and different way of approaching the whole problem. Again, forget the correct grammar, good English, and mistakes to avoid. Reach the point where you can "talk on the computer" or paper, if you insist. Write like you talk!

We've already established the fact that you are comfortable using terminals and PCs. So why not continue to use these tools to develop documents? If you develop online documentation, you must use these tools. If you still need hardcopy documents and online narrative, establish guidelines and use the computer to write. Sorry, I meant to say "write like you talk."

The trick is to remind yourself to write in a conversational tone, and in informal surroundings. Try to imagine yourself talking to one reader in person. After all, your reader is alive, hopefully well, and likes talking in the everyday conversational mode. All the ones we work with certainly do.

When most people read, they have a voice inside them speak to them. Write assuming that your reader will read your documentation the same way. How many of you enjoy talking? (If necessary, be honest). Writing can be just as easy and fun as talking. Write as you talk. You'll be surprised at the quality of the finished document if you use a conversational rather than a literary style.

You needn't impress a user or maintenance programmer with big words, technical jargon, and a barnyard full of bull. Chances are they won't be impressed. So, write like you talk. Go over what you've written and then listen to it. If it doesn't sound like talk, change it.

Rudolf Flesch has written an excellent, short, easy-to-read book on "How To Say What You Mean in Plain English". In his book, he describes seven ground rules (specific style devices) that will make your writing look and sound like spoken English. By the way, these rules may apply to all writing, not just systems documentation.

I've modified these rules and included a few of my own which will assist you in developing documentation the easy way. Here they are:

1. Use active voice.

2. Use simple whole declarative sentences.

3. Use short words.

4. Use short sentences.

5. Limit paragraphs to only two to six sentences.

6. Use contractions like it's or doesn't.

7. Use direct questions.

8. Use the pronoun "you" as much as possible.

9. Use bullets.

Are you starting to get the drift of what I'm talking about? Let's look at some of the rules in a little more detail.

Use Active Voice

The active voice expresses action. It is something which the doer does. This example is from an ad in one of the Washington papers.

"I will listen to you for $5.00 an hour. Call :::-:::::."

Here's one from one of our user documents. "Function keys 5 (left) and 6 (right) scroll the columns to the left and right for you to view."

Be specific - eliminate all generalities.

"The world is round."

See how the active voice "I will listen" and "for you to view" are action packed statements. No generalities in any of the above statements. Easy for you to write. Easy for the reader to understand. You've got to accent the conversational, eliminate the jargon, and don't mess with Mr.. In-between.

Use Simple Whole Declarative Sentences

Remember the old saying, "Tell it like it is?" The sentence declares itself. So easy to write, yet so powerful. Use it regularly.

You might remember back in the 60s when President Kennedy made a famous speech and talked about development of the space program. He said something like this. "Before this decade is out, we shall put a man on the moon." This is simple, but short. Everyone knew exactly what he said. A recent cartoon in the Washington Post updated this quote like this, "Within this decade we will send a man to the Soviet space center and bring him back."

Here's how a famous ad could have been written. "The man who is in possession of one should be asked." Fortunately, it was put in both the

active voice and in a declarative sentence or it wouldn't have become famous. "Ask the man who owns one."

So tell it like it is! Again easy to write -- easy to understand. If you only knew this kind of writing is acceptable, wouldn't you use it? Sure you would.

Use Short Words

So why do people use long words in writing? There are many reasons, but who cares? Just don't use them. Long words and technical words come between the writer and the reader. If you want your reader to understand you, then use short words. Besides, you probably won't have to worry about spelling a short word. The user won't have to look it up in a dictionary.

Here's some do and don't use words for system documentation:

DO USE	DON'T USE
yes	affirmative
no	negative
help	assist
follow	comply
send	forward
ask for	request
end or stop	terminate

Use simple, short words whenever you can. Do you think you can argue with this technique? Probably not.

Use Short Sentences

The average person can only read so many words at a time before their eyes give up and take a short rest. Newspaper writers and editors try to keep the number of words in a sentence down to plus or minus 20. It's so much simpler to construct a short sentence than a long complicated one with semicolons, colons, parentheses, and etc....

All you need to do is make short sentences with an occasional comma and a period or question mark. You will love it and so will your user. Not much grammar to worry about. Mostly a subject and a verb. And yes, it is possible to put a preposition at the end. We use prepositions at the end when we talk. No one has had to go directly to jail - do not pass go - do not collect $200 for putting a preposition at the end.

Here's a good example of a poorly written long sentence from a real life document I received. A total of 46 words!

"In consideration of this implication, our first and fundamental recommendation concerning the future HQs supplied system (software) end user documentation is that it and the locally developed and maintained accounting system documentation each acknowledge and compliment the information provided by the other, both conceptually, and logistically."

Why not just write - "Our first recommendation is ::::::::::::::."? Shortening your sentences is one of the best and easiest ways to improve your writing style. Surely you can use the short sentence technique.

Keep Your Paragraphs Short

If using short words and short sentences make sense, why doesn't the same logic apply to paragraphs? It does.

Here's an excerpt from a document I received. A good example of how not to write a paragraph with only one sentence. By the way, this has 52 words in it.

"In other words, are sufficient Human and material resources available (commitment) to prepare and/or update the Documentation at the same time that particulars of its subject change, and does the method of communication of the Documentation material (logistics) support rapid and reliable dissemination to the targeted End Users of the Documentation."

Now don't ask me what that paragraph says. Because, I don't know. You understand what I am saying. Keep your words short. Keep your sentences short. Keep your paragraphs short. Easy to write. Easy to read. Any of us can write this way.

Use Contractions

There are fifty or so contractions that are commonly used in writing. I don't want to take the time to list them all. A few examples are: I'm, you're, he's, she'll, aren't, haven't, won't, don't, and let's. I've used a plenty of them in my presentation.

Don't get weird and invent your own like it'd and this'll. Stay with the accepted ones.

So, use contractions because that's what we use when we talk. You probably use them too.

Use Direct Questions

There's nothing like a direct question. Use them freely, especially in preparing online documentation. Do you want to continue? Do you want to edit your input? Do you need help? All the user has to do is type in a Y or N. Keep the questions short and direct. Let the user answer as simply

as possible. There's nothing like a direct question to get some feedback to what you are saying.

Use Pronouns As Much As Possible

When you're talking to people, don't you use I, me, you, we and they? Sure you do. Why not use pronouns when you are documenting? I love the word "You" when you are explaining something to the user or maintenance programmer. Consider the following examples:

- o You do this.
- o You enter.
- o To help you.

How about using this techniques as it makes documentation personal and warm. You are talking directly to one person. Only one person - you.

See how easy it is for you to talk as you write?

Use Bullets

Although you don't talk using bullets, I like to use this technique in documenting for the following reasons:

- o It makes writing easier.
  - Phrases, statements or sentences may be used.
  - Little or no punctuation is used.
  - Statements can be broken down to at least four levels.
  - Not much grammar to worry about as you don't even need to write complete sentences.
  - No numbering scheme is necessary; e.g., 1.a, 1.b, etc.....
- o It makes reading easier.
  - The facts aren't all jammed together in a paragraph.
  - It's easy to make notes or references to a line item.
  - You can easily concentrate on one line at a time.

See how clean this looks and each line jumps out at the reader. If you need to go to lower levels, here's how we break it down:

o level one

- level two

oo level three

-- level four

This technique is easy for you to use, and it makes your documentation easy for the reader to read.

ONLINE DOCUMENTATION

We need a commitment to user-friendly communications: online documentation.

So much for tradition. Let's make it easy for us and our users. Let's start developing online operational documentation for the mainframes, the HPs and the microcomputers (PC) systems. The PC folks have been doing this for years. Why not strive for a paperless society?

Did you notice the statements on the income tax instructions about form reduction? What a laugh! And, their instructions are something else, aren't they? The instructions are user-nasty. The IRS hasn't really helped us this year. To illustrate, here's a cartoon from the Washington Post:

"Dear Tax Accountants:

I'd like to get your help in figuring out the new tax instructions and this year's tax forms. They've really got me stumped.

Sincerely, John Q. Jones IRS Agent Treasury Dept.... Washington, D.C."

Well, let's not be like the IRS. Go for online documentation. You can still have some hardcopy documentation or Quick Reference Cards available. Or even tell your users they can print a screen if they need to for a paper reference. Isn't this much easier than looking something up in a three-inch documentation manual? Sure it is.

We've established that tradition is out, and you can write like you talk. And, you can develop documentation in your own environment using the skills and tools you already have. Now we'll talk about some methods to provide online text. This subject could be a technical discussion all by itself, so I will only touch on it lightly.

Tutorials

You should provide the user with tutorial assistance screens. This provides first-time users and maintenance programmers with the basic information on "how to use the system". This is also a valuable tool for initial and ongoing training seminars. Tutorials are also good for describing screens and error messages. Remember to write like you talk.

Screen Design

Very briefly, design the screens to help the user. Design them to create a paper document if necessary. Provide user guidance and prompting assistance. Highlighted movable lightbar (block cursor) prompting is great. If you have the luxury of color monitors, take advantage of using color. Highlight errors and provide a brief description at the bottom of the screen.

Online menu screens provide prompts and help. They should provide the ability for the user to request more detailed online help. Develop screen guidelines to use for screen consistency.

Help Prompts

Context sensitive help is provided through alternate PF key selection. Offer help prompts on all level 1 screens. Briefly define the prompt at the bottom of the screen; for example, 1=help, 2=next, 3=previous, 4=create, 5=report, and 12=quit. Try and keep the PF keys consistent within a system whenever possible. For more user assistance, help overlay or level 2 screens may be initiated by pressing the PF keys designated on the screens.

System Maintenance Directory

The hardcopy segment of this programmer's document is truly a directory. It serves as a road map to the entire online system. Why shouldn't this document be online for the maintenance programmer? Sure it can. Store your inventory of programs and files online for easy updating by the programmers. All narrative can also be online for easy browsing and maintenance.

Internal Program Documentation

Every program developed must have comments within the program to document it. If you aren't doing this - please start immediately!

At the beginning of each program, there should be a description of the function performed by the program. Include all files used the by program, data sets accessed or modified, and all entry points. Also, all sections that perform major functions or contain complicated code should be explained by comments throughout the program.

A log of all changes to a program should be kept at the beginning of the program for future reference. This log must include the date of modification, change or problem request number, initials of programmer making change, and the reason for change.

The use of new 4GL languages heightens the importance of internal documentation. COBOL has been with us for a long time. Most programmers are experienced with it and consider COBOL reasonably straightforward and easy to follow. C language, for instance, is full of peculiarities and may be hard to follow if it isn't carefully documented.

For all of our mainframe CICS applications, we have developed standard front-end programs and screens. These include a system entry program with administrative routines; for example, date conversion. Other examples include a message file, a system owner file, and a tutorial broadcast file. Each of these are defined online for the programmer's use. The programmer can then concentrate on the specifics for the application system. We even have a hotline report to indicate user problems. The report contains such information as abends, error message numbers, and user terminal. The hotline programmer can give quick response in helping the user.

In summary, online text is more easily modified and maintained than paper documents, and may be used to create paper documents if needed. Online screen displays are easier to illustrate, and the text can provide the user with online assistance and reference capabilities. Tutorials lead first-time users through the basic job tasks by means of a series of exercises. And, you can develop this documentation as you program.

CONCLUSION

This technical session has provided you with some practical guidelines. I trust it has stimulated an awareness of how you can provide quality, easy-to-use operational system documentation.

Documentation methods are giving us new power to improve the state of our documentation. These techniques and tools will help to make it easier for us to create better documents, and of course provide better products - but, only when our attitude towards documentation changes. This includes us, our staffs, and our users.

I would like to challenge each of you to continue to develop innovative methods for producing quality documentation. Use methods that integrate online and hardcopy media in a cost-effective manner throughout your systems life cycles.

And even more important, from a technical perspective, why you can grow up to write online operational documentation. And, you can enjoy it!