

The Software Development Executive's Guide to Managing Technical Publications

Technical publications include a wide variety of materials, such as user guides, online help, white papers, classroom training materials, reference guides, administrator guides, standard operating procedures, tutorials, installation and configuration guides, computer-based training, and web-based training. In most organizations, the product development executives bear ultimate responsibility for these deliverables. But expertise in managing development of core products (such as software applications or machinery) doesn't always translate to familiarity with product documentation issues.

This overview of technical publications describes documentation deliverables and staffing options. If you are an executive who has not previously handled technical documentation responsibilities, this white paper will give you enough information to get started.



Scriptorium Publishing Services, Inc.

P.O. Box 12761, Research Triangle Park, NC 27709-2761

919-481-2701 or sales@scriptorium.com

<http://www.scriptorium.com>

Contents

The purpose of technical documentation	3
Getting information to the end user	6
How documentation is used—and what to do about it	7
The trouble with technical documentation	8
Documentation options	9
Staffing options	18
Choosing a staffing option	22
What to look for in a technical writer	23
Coordinating software development and technical publications	26
Where to go from here	27

The purpose of technical documentation

In the real world, technical documentation is often developed as an afterthought. This usually results in inaccurate, hard-to-understand materials. Not surprisingly, users dislike these materials and avoid using them; instead, they turn to alternatives, such as customer support hotlines or user groups. They may reject the product because they can't figure out how to use it.

When you deliver a product—whether a software application, a tractor, or a computer-controlled nuclear waste incinerator—to customers, you must also give the users information about how to operate the product. *The purpose of technical documentation is to transfer product knowledge to end users.* Depending on the type of product you create, you may need to provide several levels of information and use multiple strategies to communicate the information that's needed. To use a product successfully, users need at least two different types of knowledge:¹

- *Domain knowledge* is information about the subject matter. If your product is a computerized paint-mixing machine, domain knowledge would include color theory. To use the machine successfully, end users need to know basic color theory. Domain knowledge might also include interior design knowledge—it would be helpful to know how to create a coordinated color palette.
- *Product knowledge* is information about how to use your product. After choosing a particular color based on domain knowledge, users would need product knowledge to configure the machine so that it creates that color.

To ensure that your documentation is useful, you need to figure out how much domain and product knowledge your end users have when they get started and how much more they need to use your product successfully.

Figure 1 shows one way of visualizing the knowledge transfer requirements.

1. Jared Spool of User Interface Engineering (www.uie.com) has developed many of the concepts discussed in this section.

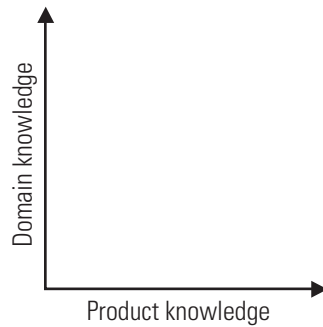


Figure 1: Graphing domain and product knowledge requirements

Writing for domain experts

If your end users start with extensive domain knowledge, they need to learn how to use your product but do not need background information about what the product is doing. For example, consider an intelligence analyst who is accustomed to reviewing satellite imagery on paper. Switching this analyst to an online system would involve learning how to use the software magnification instead of the old-fashioned magnifying glass. You would not, however, need to provide instructions on what to look for in the images—the analyst already has that domain knowledge. Figure 2 shows the knowledge needed by the analyst; you would need to provide the information needed to move from A (high domain knowledge but no product knowledge) to B (high domain and product knowledge). To provide documentation for domain experts, you focus on your product's features.

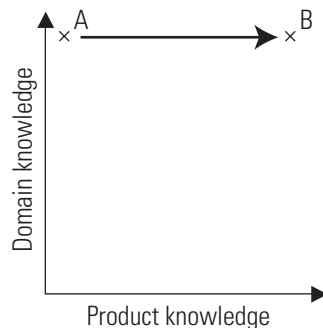


Figure 2: Domain experts just need product-specific information

Writing for novice users with no domain knowledge

Users who start without domain or product knowledge present a more difficult problem. Consider, for example, someone who is using a database product for the first time (user C in Figure 3).

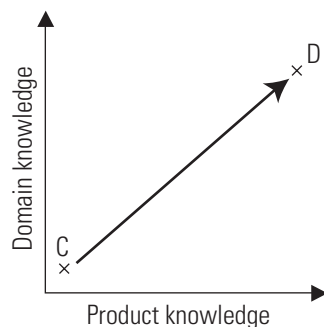


Figure 3: Working with databases requires domain knowledge and product knowledge

Before explaining how to create a table and assign a primary key, you have to provide some basic information about how databases work and define the terms table and primary key.

For a database administrator (the domain expert), you could omit the database concepts, but if your users are new to databases, you must provide domain knowledge because they cannot use your database software without it. The new database user needs lots of information.

Reducing domain-knowledge requirements

It's possible to minimize the domain-knowledge requirements with the product's design. Microsoft Access, for instance, can automatically assign a primary key in a table—the user doesn't need to know it's happening. The paint-mixing machine could include recommended color groupings. Figure 4 shows how building domain information into the product reduces the amount of knowledge needed.

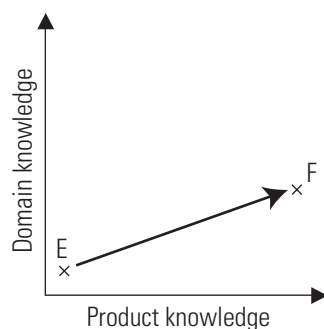


Figure 4: You can reduce domain-knowledge requirements by building the information into the software

Considering user motivation

If you require that users invest a lot of time in learning about your product and learning new domain knowledge, your product must offer users compelling benefits. Consider the difference between home image-editing programs (used with digital cameras) and a professional application such as Adobe Photoshop. The home programs are more limited but are much easier to use. They have features (such as “remove red-eye”) that correct common problems in photographs. In Adobe Photoshop, by contrast, eliminating red-eye is much more difficult, but users have more control over the process. The two applications are intended for different audiences. The casual user wants an application that’s easy to use; the professional user prefers the application that offers the most control over the final image. Figure 5 compares the learning required for the two types of users.

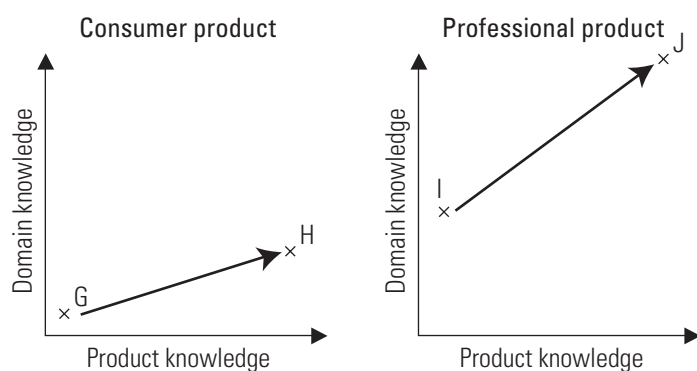


Figure 5: Consumer-level vs. professional product

The different needs of your customers and the gap between their starting and ending points determine how you should organize your knowledge-transfer efforts. The purpose of technical documentation is to make it possible for end users to use the product successfully.

Getting information to the end user

Technical documentation is only one way to provide information to the end user. Phone and email-based technical support is another option. But technical documentation lets you create the content once and then use it for many different end users. Providing help desk and other one-on-one support is very expensive (Figure 6).

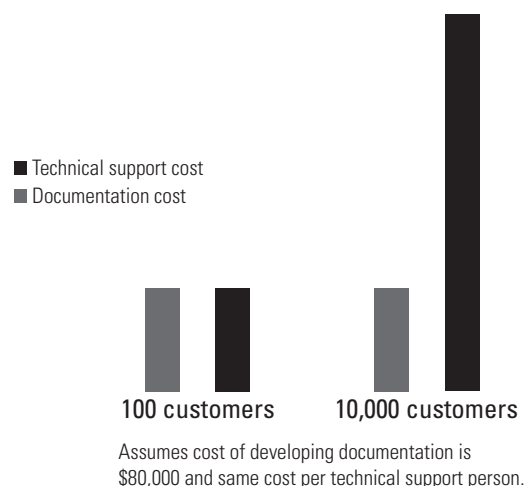


Figure 6: Cost of technical support and documentation development (doesn't include printing costs)

Because technical documentation allows you to produce a single set of materials and then distribute those to all your customers (one to many), it's much less expensive than providing technical support (one to one). Providing good documentation reduces support calls because users can look up information on their own. Charging for technical support can reduce some of the cost, but it's quite difficult to charge enough to cover the costs of the technical support organization. Furthermore, the technical support staff needs to know how to use the product—which means that you need documentation for them.

How documentation is used—and what to do about it

Most users don't read documentation until they have a problem, so typical documentation use goes something like this:

1. The user runs into a problem using the product.
2. The user tries to work around the problem in a few ways.
3. The user looks either in the online help or the printed documentation for an answer. Some users start with the index; others start with the table of contents.²
4. The user finds an entry that looks relevant and checks that part of the documentation for the information.

2. A tiny percentage of users read all of the documentation before using the product.

5. If the answer is found, the documentation “passes.” Next time the user has a problem, the documentation gets another chance.

If the user doesn't find the information after checking two or three entries in the book, the user gives up and calls tech support. Furthermore, this user probably won't consult the documentation next time there is a problem. This user now believes that the documentation is useless.

You can draw some interesting conclusions from this simple scenario:

- If end users cannot find the answers to their problems, they assume that the documentation is useless.
- Providing useful, accurate indexes and tables of contents is critical to ensure that users can find information.
- Very few users read documentation in detail or sequentially, so it's important that sections stand on their own and can be read in chunks.
- Documentation must prove itself the first few times it's consulted; otherwise, the user learns not to use it and switches to other means of getting information.
- 90 percent of the users read only 10 percent of the documentation. Unfortunately, each user reads a different 10 percent of the documentation.

The trouble with technical documentation

To be useful, the documentation must be clear, concise, accurate, and generally well assembled. The trouble is that a lot of technical documentation is just plain bad. Recognizing poorly written, badly organized documentation isn't difficult, but preventing it can be.

Occasionally, you'll encounter documentation that is technically accurate and complete but poorly designed. People *do* judge the book by its cover—they assume that the interior must be unusable. In an extreme case, the end user might never even open the book and just skip directly to technical support because the book looks bad. The small percentage of extra time and money spent to ensure that documentation looks good is well worth the effort.

Producing attractive documentation is a useful marketing tool, especially for small companies. A large company has name recognition, which can mitigate the damage done by poor documentation. But if a small, unknown company produces material that doesn't look professional, potential clients may assume that the product must not be particularly well designed, either. Therefore, they reason, the product itself probably doesn't work. Buying the small company's product is already considered a bigger risk than buying from a large, established company. The documentation can reinforce or help reduce that impression.

Documentation options

Technical documentation is any information product that transfers knowledge to the user. This could include any or all of the following:

- User guides
- Administrator guides
- Reference guides
- Online help, context-sensitive help (interface help), ToolTips, and error messages
- Training materials, such as instructor guides, student guides, and hands-on exercises
- Computer-based training
- Web-based training
- Knowledge bases and frequently asked questions (FAQs)

The following sections provide brief descriptions of these products and their uses.

User guides

User guides describe how to perform common tasks. They are aimed at entry-level or intermediate users. A user guide for an email application, for instance, would describe common tasks such as sending and receiving mail, composing messages, and attaching files to messages. Figure 7 shows a typical "how-to" section from a user guide.

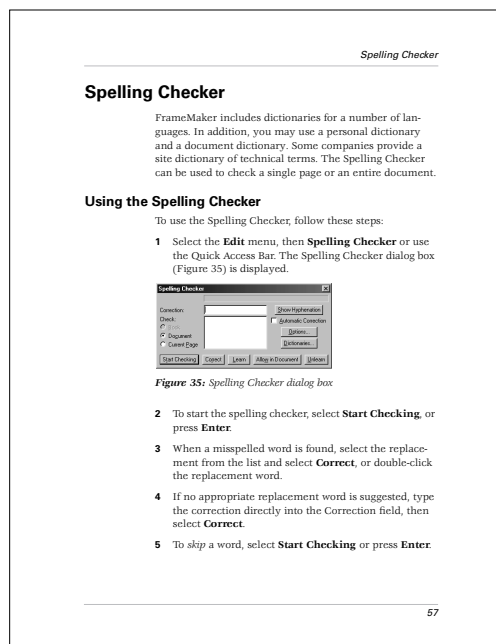


Figure 7: A typical section in a user guide

Administrator guides

Administrator guides describe configuration tasks that are usually done by only a few advanced users and not by the “general” user. For an email application, the administrator guide would describe how to configure the mail server and how to set up new user accounts. Administrator and user guides have a similar “how-to” approach, but the information in the administrator guide is more technical.

Reference guides

Reference guides (Figure 8) describe functions available in a system. Documentation for programming languages usually includes reference guides. Most often, reference guides are organized by command in alphabetical order. Because reference guides provide detailed information, they are popular with more advanced users who want to customize an application to their specifications.

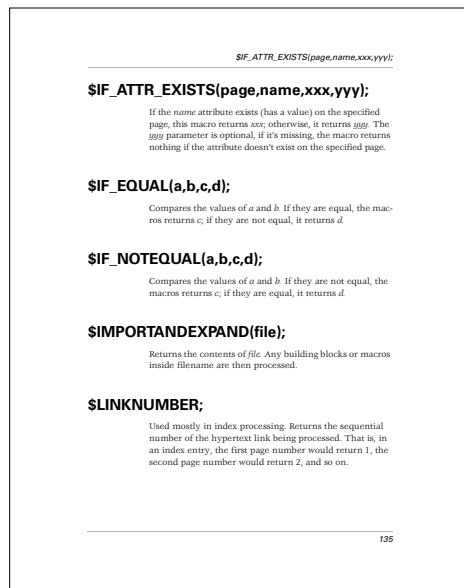


Figure 8: Alphabetical listings in reference guides make information easy to locate

Online help

Online help (Figure 9) refers to information that users can display from a software application's Help menu. Online help provides quick instructions on how to complete a task. The information may be similar to the information in the printed documentation. Some users prefer to use online resources; others prefer printed books.

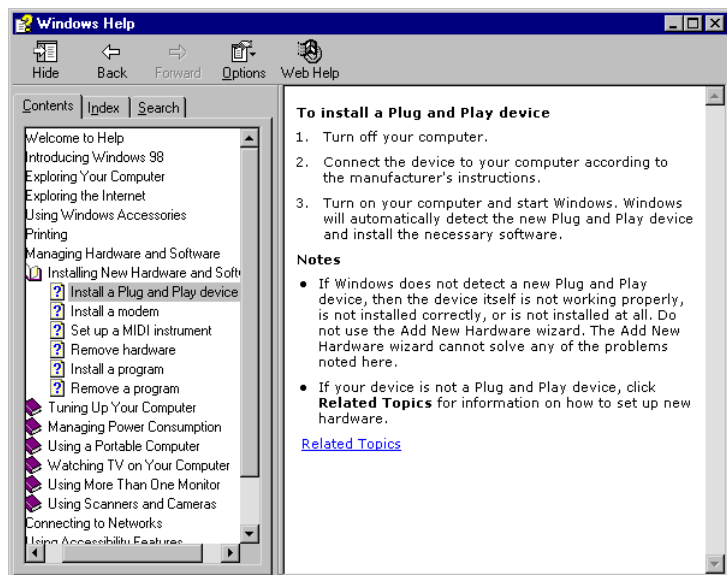


Figure 9: Windows 98 provides quick how-to information

When you create online reference material, you can take advantage of hypertext linking and full-text search to provide information that is much easier to search than the printed version. One common example of this would be a glossary. In printed books, the glossary usually appears at the end of the document, just before the index. This forces the reader to flip back and forth to find definitions for terms. In online help, you can set up the glossary definitions as pop-up topics, so that a user can click on a glossary word in a topic to display the definition (Figure 10).

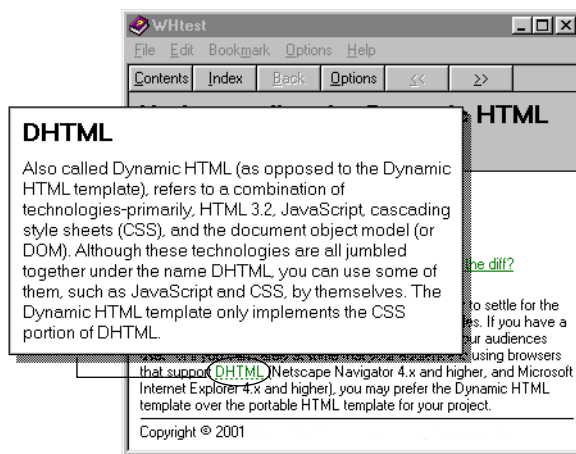


Figure 10: Making an interactive glossary entry—the term DHTML is linked to a definition

Most help technologies offer a way to provide *context-sensitive help*. This means that the information displayed when the user requests help depends on what item in the interface has been selected. Context-sensitive help is time consuming to implement and requires close cooperation between the developer and the technical writer. However, it's very popular with the end user because it provides exactly the information that's needed (Figure 11).

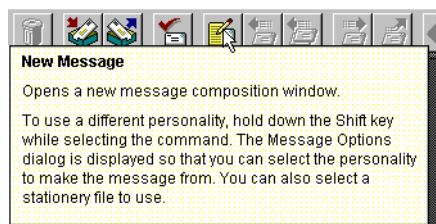


Figure 11: Just-in-time information from context-sensitive help (Eudora Pro email application)

Adding online help and context-sensitive help to the list of deliverables requires that you hire technical writers who understand how to create online help. Furthermore, they need to understand the various online help technologies that are available, such as HTML Help, Help 2, JavaHelp, InterHelp, WebHelp, and browser-based HTML help. There are a number of different tools available for creating online help; it's also helpful to look for a writer who understands *single-sourcing* concepts. Single sourcing refers to a documentation approach in which content is written once and then used in several different output formats. This usually involves writing scripts to convert from one format to another. Figure 12 shows a single-sourcing method based on Adobe FrameMaker files.

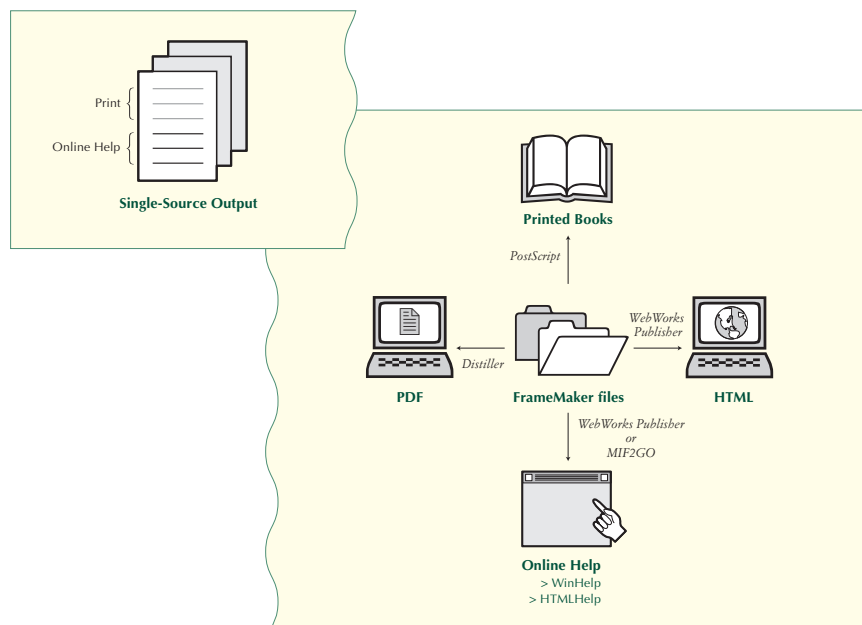


Figure 12: Single sourcing can improve your staff's productivity

Training materials

Training materials (Figure 13) often build on user documentation with the addition of exercises, tutorials, and other hands-on instructions. Course developers create sample scenarios, which allow trainees to learn how to use the product in a safe environment rather than on the job. (An extreme example of this would be the multimillion-dollar flight simulators for training commercial airline pilots—it's better that they make flying mistakes in a simulator!)

Printed training materials usually include most of the following items:

- The *student guide* describes all the information that the student learns during a class. Student guides are similar to user guides.
- The *instructor guide* describes what topics the instructor needs to cover during the class, the objective for each module, and any special instructions, such as demos to conduct and tips to remember. Depending on the skills and experience of the instructors, the instructor guide could be brief (a bulleted list of topics) or quite lengthy (an explicit “script” that gives the instructor a complete road map for the class).
- *Exercises* provide step-by-step instructions for particular tasks. For example, in a class on using email software, an exercise might provide a walk-through for replying to a message or adding an individual's address to the address book.

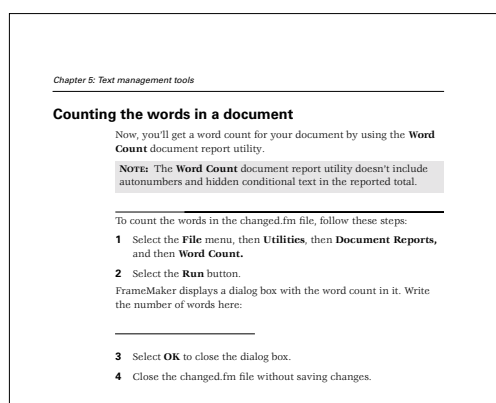


Figure 13: Training lets users try out new concepts

Computer-based training

Computer-based training (CBT) is stand-alone training, often run from a CD, that eliminates the need for an instructor. A workbook may accompany the CD. In a traditional instructor-led class, the instructor provides conceptual information in a lecture. In CBT, the instructor is replaced by video and multimedia clips that communicate the conceptual information (Figure 14).

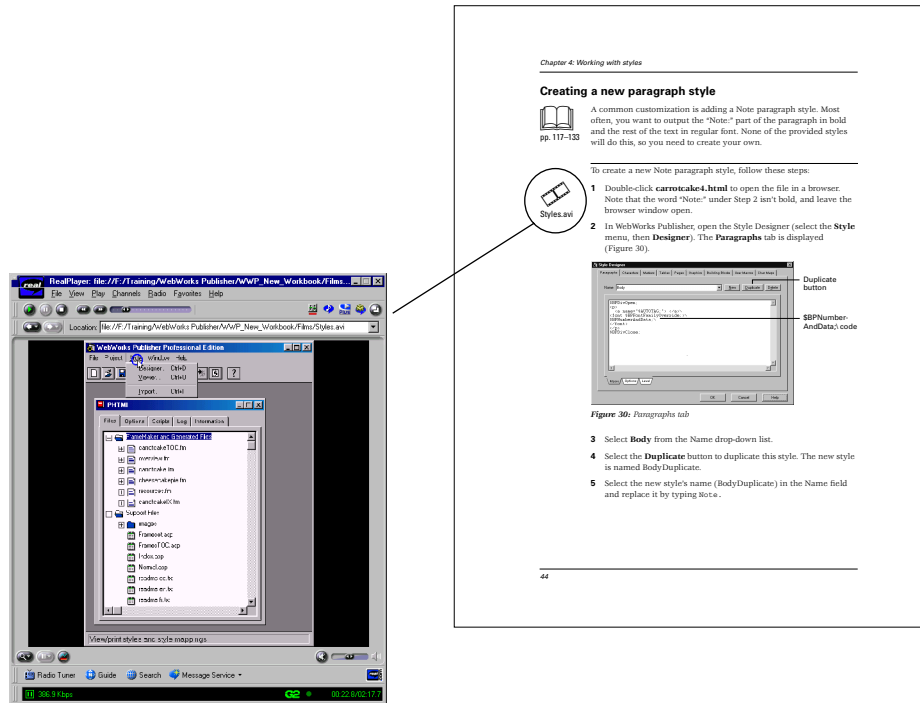


Figure 14: Computer-based training brings the classroom to your screen—the workbook instructs the student to watch a video clip

Instructor-led training is preferable because the student-instructor interaction provides immediate feedback and the opportunity to ask questions as they arise. But if you need to train a large number of people, then CBT is much less expensive than instructor-led classes. The cost to deliver CBT to 10 people is basically the same as for 1,000 people; you just create more CDs (see Figure 15). For instructor-led training, the cost increases exponentially because you must pay for an instructor's time in each class.

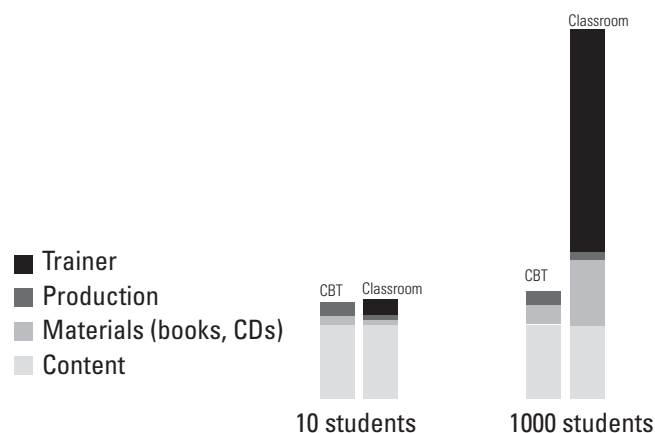


Figure 15: Sample cost analysis for CBT and instructor-led training

Web-based training

Web-based training (WBT) is stand-alone training like CBT. But WBT introduces yet another challenge into the development process—the bandwidth limitations of working with web-based content. WBT uses many of the same techniques as CBT, but often the multimedia components are streamlined so that they are easier to deliver over the web. Because information is stored on the World Wide Web instead of being distributed on a CD, WBT is easier to update. If the content of the training changes frequently and it's critical that students see only the latest version, WBT provides an easy way to control which version students see.

Knowledge bases & frequently asked questions (FAQs)

Knowledge bases and frequently asked questions (FAQs) are similar, but FAQs are a little less formal than knowledge bases. In either case, you provide information to your end users, generally over the web. The users can search a database for answers to their questions. In many cases, companies include questions that help desk personnel answer frequently. The knowledge base is a little like a self-service technical support hotline—if the user can formulate the right question, the answer is close at hand (Figure 16).

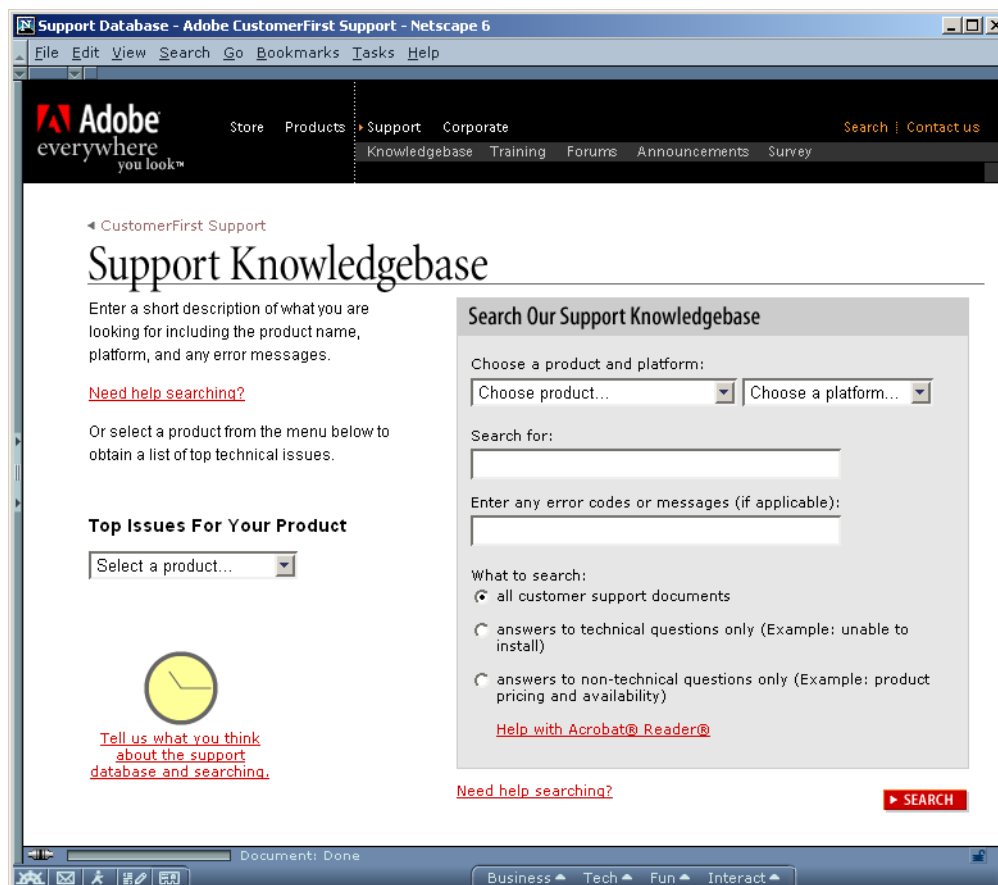


Figure 16: Adobe's knowledge base (<http://www.adobe.com/support/database.html>)

Knowledge bases are appealing because they are much more efficient than one-on-one technical support. It's critical, though, to have thorough indexing and search utilities so that users can find the information they need. Knowledge bases also need to be maintained and updated to ensure that the information doesn't become outdated.

Creating knowledge bases requires some specialized skills from the technical writers. They need to understand information-retrieval techniques. For example, each chunk of information in the knowledge base needs to be labeled with relevant keywords so that searches turn up the right information.³

3. Each chunk should also have some associated *metadata* (information about the information), such as the date it was updated and the platform and software version for which this chunk is written.

Staffing options

There are several different staffing options for your project. The size and scope of the projects should determine which one you choose. Here are your choices:

- Use programmers
- Hire part-time or full-time staff writers
- Hire freelance writers
- Outsource to documentation company

For the last three options, this document assumes that you hire qualified, professional technical writers. (See page 23 for some tips on how to evaluate technical writers.)

Let the programmers do it!

Given time constraints and limited resources, many companies are tempted to “just let the programmers do it.” After all, they reason, the software developers know the application better than anyone else, so why not have them “whip out” some documentation.

Here are a few faulty assumptions that result in problems with the documentation when using this approach:

- *The developers know the application better than anyone else.* The developers know the application *too* well. The end users only want to know how to get a particular task done—they rarely care about the elegance of the underlying code. The developer is likely to provide technology-centered information; the end user wants task-centered information. It's almost impossible for the developer to step back and look at the application from the end user's point of view. See Figures 17 and 18 for a comparison of how developers and end users view applications.

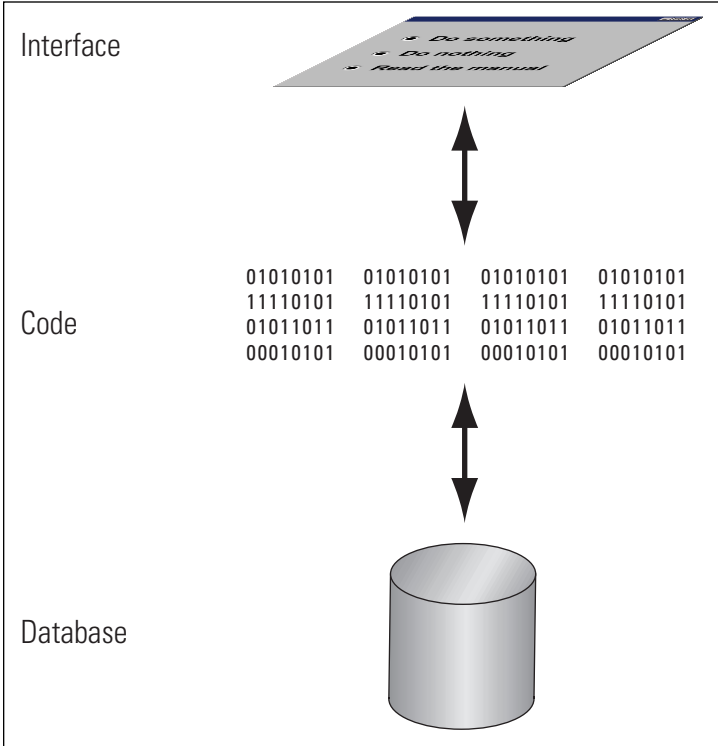


Figure 17: How the programmers see the product

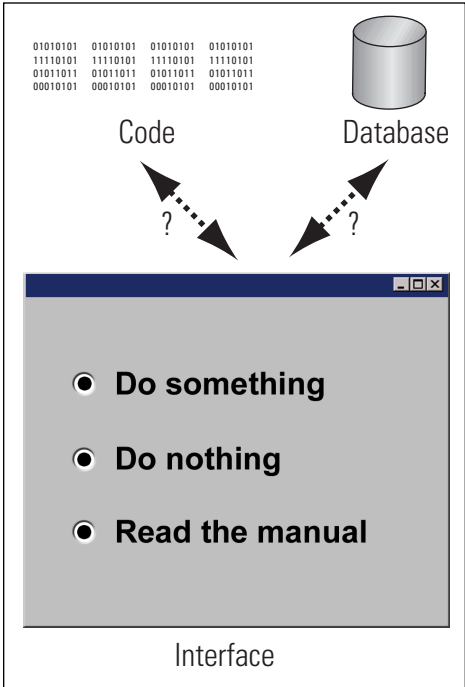


Figure 18: How the user sees the product

- *The application is complicated; having a writer learn about it will take too long.* Technical writers spend much of their career learning about new applications. If you hire a good technical writer with the right background, learning your application should not be a major obstacle. If your audience has specialized knowledge, consider hiring a technical writer who already possesses this knowledge. For instance, if your product is a computerized DNA analysis tool and the end users are all biochemists, the writer probably needs a biochemistry background.
- *Programmers can write documentation just as well as technical writers.* Programmers spend their days writing code. A few programmers are also good writers, but there aren't very many of them. For some programmers, there is also a language barrier. Technical writers spend a lot of time learning how to structure documents, how to put together information to make it easily comprehensible, and other writing "tricks." The programmers, even the few who write well, do not know about these tricks. The professional technical writer is more efficient and produces higher-quality documentation. Furthermore, having the programmers write documentation takes them away from their primary task—creating the software. This productivity hit on the software development effort needs to be factored in if you want to use programmers to write documentation.

Using programmers to write documentation is likely to result in poorly written, product-centric materials.

Staff writers

Staff writers, or employees, provide you with dedicated in-house resources. The major advantages of employing staff writers are:

- They're always available.
- Over time, they develop in-depth knowledge of the company. Because you have the same people developing documentation over several years, they can build a deeper understanding of the company and the big picture.

If your company has an ongoing need for documentation, a documentation group, technical writer, or perhaps a part-time technical writer may be the best solution for your requirements.

The disadvantage to staff writers is that they become a permanent part of your staff and of your expenses. If your technical writing requirements are sporadic, having employees who are dedicated to this task may not be the best solution.

Using a staffing agency lets you adopt a “try-before-you-buy” approach with a temp-to-perm contract. This can be a useful hedge if you’re not certain that you can evaluate technical writers during an interview.

Freelance writers

Freelance writers are generally experienced writers who take on a few projects each year. If your documentation requirements are limited, a freelancer can be a good—and usually inexpensive—solution. One major disadvantage to working with freelancers is that they tend to work alone. If they are already committed to another project when you call, they may turn down the work. And if your project requires more than one person, you may have to coordinate several different freelancers.

Freelancers work best when you have a project that is well defined and that can be completed by one person in a specific time frame.

Documentation companies

Documentation service providers are generally small companies whose staff includes all the different functions you’d expect in a documentation group—writers, editors, template designers, and production specialists. You can use documentation companies to outsource your entire technical documentation effort.

This approach makes a lot of sense if your company meets one of the following scenarios:

- You don’t want to hire in-house staff for documentation because you want to concentrate on core competencies, such as software development.
- The workload for any technical writer fluctuates greatly, and any in-house staff would be either overwhelmed or have no work at all. This is common in small companies, where documentation development is needed for only about three months before each release, and releases are one year apart.
- You have some work that requires senior tools specialists and other work that can be completed by junior writers. Most documentation companies will staff the project accordingly,

so you'll pay less for the junior writers than for the high-end consulting.

A professional documentation company can provide just-in-time services that replace your in-house documentation department and offer higher throughput than a single freelancer.

The disadvantage to working with a documentation company is cost. The hourly rates will be higher than what you'd pay a staff writer. So if you have ongoing work, an in-house hire will likely save you money. But if the work is intermittent, a documentation company is a good choice.

Choosing a staffing option

The staffing you choose will depend on the weight you place on a number of different factors (Table 1).

Table 1: Evaluating your documentation options

Issue	Options
Preserving corporate knowledge	<p>Making a commitment to staff employees will result in lower turnover and thus better retention of corporate knowledge within the company.</p> <p>If staff employees are not an option, establish a long-term relationship with a freelancer or with a documentation company that has low turnover.</p>
Maximizing quality	<p>The quality of the finished product will depend entirely on the skills of the individuals producing the documentation. Any of the staffing options can provide you with skilled staff; any of them can result in poor quality. If you do not feel comfortable evaluating the quality of technical publications, pick a project manager whom you trust.</p> <p>If you do not want to establish an in-house documentation department, consider hiring an experienced manager as a staff employee, and having that manager coordinate outside vendors.</p>
Minimizing cost	<p>You can minimize costs by hiring inexperienced writers, but you put the documentation quality at risk.</p> <p>You can also reduce total cost by staffing as needed; typically by using a freelancer or documentation company that is paid only when working on your product.</p>

Table 1: Evaluating your documentation options (continued)

Issue	Options
Workload fluctuates	Hire a core staff and augment it with freelancers or a documentation company during busy times. Alternately, outsource the entire documentation effort and let the documentation company figure out how to manage the staffing requirements. See Figure 19.
Workload is constant	If the workload is constant, in-house staff is generally more cost-effective than outsourcing.
Don't want to manage technical writers	Outsource to a documentation company that provides strong project management and that keeps you informed.
Focusing on core competencies	Outsource.
Maximizing technology to improve productivity	Implement the specialized tools and technologies that make the writers more efficient. Either hire specialists or train staff writers.

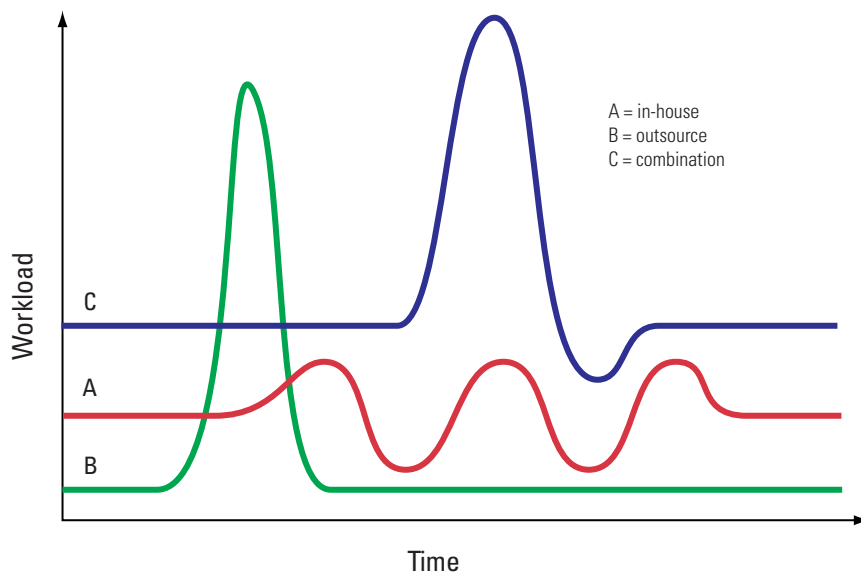


Figure 19: Some companies experience major fluctuations in their documentation requirements

What to look for in a technical writer

Although the exact skill set you want in a technical writer varies somewhat depending on whether you plan to hire or outsource, there are some common characteristics you should insist on in any writer. They are:

- Writing ability
- Knowledge of technology
- Knowledge of tools
- Personal traits

Writing ability

A technical writer must be an excellent writer. Unfortunately, this critical skill can be difficult to evaluate. During an interview, ask for writing samples. If you don't feel comfortable evaluating writing, ask a colleague with good writing skills to check the samples. Technical writing should be clear, concise, and grammatically correct. The first writing sample you'll receive will probably be a resume. Look for typos, grammatical errors, and other mechanical problems. If the writer cannot produce a clean two-page resume document, odds are none too good for longer material.⁴

During an interview, probe for writing skills by asking the writer how he or she checks drafts for errors. There is no single correct answer; listen for evidence that the writer has thought about the issue and has some techniques to ensure that drafts get corrected.

Technology

The writer should be able to understand the technology that he or she will write about. If the writer is going to document how to use an application programming interface (API), some experience with programming and the ability to read code would be desirable. If the writer is going to document a database driver, knowledge about database concepts would be useful, although the writer doesn't necessarily need to have experience as a database administrator.

When evaluating a writer's resume for technology, look for experience in fields that are related to your product. For example, experience in C++ would translate well to working with Java. Consider also how technical the technical writer needs to be. Consumer software, such as email applications and word processors, requires a writer who can write very simple instructions for people who are not particularly comfortable with computers. Writing documentation for a programmer's integrated

4. Scriptorium Publishing's standard is to allow one typo per resume. Two or more typos, especially if the candidate claims to having editing experience, results in the candidate being rejected immediately.

development environment (IDE) requires a writer with a highly technical background. A candidate who has written documents for an audience similar to your customer base is preferable. If your audience has specialized knowledge, a candidate with a similar background is helpful.

Tools

You probably assume that a programmer who understands the concept of object orientation and knows C++ can learn Java fairly easily. Similarly, tools should not be a major obstacle for technical writers. It's helpful, however, to classify the various tools, just as you can classify programming tools (see Table 2). You can gauge a writer's level of tool knowledge by looking at the class of tools that he or she is familiar with. When hiring senior writers, look for individuals who are comfortable with lots of different tools across different groups if they don't know the one you're planning to use.

Table 2: Grouping tools by complexity level

Level	Writer
Entry-level	Word processors and office applications, such as Word, Excel, AmiPro, WordPerfect HTML
Professional	FrameMaker, Interleaf, Ventura Publisher
Specialized	SGML editors, such as AdeptEditor, ArborText Microsoft Project for project management WebWorks Publisher Perl and other scripting tools Databases
Marketing focus rather than technical focus	PageMaker, QuarkXPress, InDesign
Online help	RoboHelp, ForeHelp, Doc2Help WebWorks Publisher, MIF2GO (if converting from FrameMaker)
Graphics	Illustrator, FreeHand, Visio (flowcharts), CorelDraw, Photoshop
Computer-based and web-based training development	Director, AuthorWare, DreamWeaver with CourseBuilder extensions

Personal traits

In addition to the skills outlined in the previous sections, technical writers need a combination of soft skills that can be difficult to find. The writer needs both assertiveness and tact because a big part of writing technical documentation is extracting information from often uncooperative sources—the programmers.

Beware, however, of hiring a writer who will act only as a typist. The writer should be able to figure out the software with minimal assistance from the programmers. The programmer's role in software documentation should be limited to answering complex questions and reviewing material for accuracy. If your writer does nothing more than clean up drafts that the programmer has written, you have hired an administrative assistant and not a technical writer.

Coordinating software development and technical publications

The technical writer's job is to interpret complex information about your product in a way that makes sense to the end user. To accomplish this, the writer needs timely, accurate information about the product's development status. There are a few easy (and inexpensive) ways to do this:

- Make sure that writers attend development meetings or at a minimum are on the developers' status mailing list.
- Keep the writers informed about product changes.
- Give the writers access to the bug-tracking database.
- Provide writers with a reasonable amount of time with developers to get the information they need.
- Require developers to review documentation to ensure it is accurate and complete. Developers need not look for grammatical errors, but they should identify any technical mistakes or omissions.

Developers are often tempted to short-change the writers because of pressure to make a coding deadline. Management can avoid this problem by making it clear that developers are expected to cooperate with writers (whether staff or contract).

Most developers focus on a portion of a product or on one product in a product line. Writers usually see a much bigger slice of the products—the typical developer-to-writer ratio is about 8:1.

This means that the writer probably has a good understanding of the overall design. You can take advantage of this by using writers as interface design reviewers. Because they have to explain how to use the product, writers are often the first to identify problems in the interface design—if it's hard to explain how to use it, it can be an indicator that there's a problem with the interface.

Where to go from here

If you haven't already done so, take a close look at your product and your customers. Are your customers learning what they need to? Do they complain that your product is difficult to use? Are they flooding the technical support lines? If so, your product may benefit from improved technical documentation. Done right, technical documentation can provide a competitive advantage and pay for itself by reducing support costs and increasing sales.

If you need assistance in building a documentation group or want to outsource your documentation, contact Scriptorium Publishing.

Contacting us

Scriptorium Publishing is a leading provider of writing, editing, training, and consulting services in the technical publishing industry. We provide outsourced technical documentation services and training for technical writers.

If you have any questions about Scriptorium Publishing Services, Inc., contact:

Scriptorium Publishing Services, Inc.
P.O. Box 12761
Research Triangle Park, NC 27709-2761
919-481-2701
sales@scriptorium.com
<http://www.scriptorium.com>

All trademarks used herein are the properties of their respective owners and are used for identification purposes only.