DECEMBER 2019

intercom

THE MAGAZINE OF THE SOCIETY FOR TECHNICAL COMMUNICATION

CONTENT QUALITY

A Framework for Thinking about Documentation Quality

By STEVEN JONG | *STC Fellow*

OUR INTEREST IN DOCUMENTATION QUALITY isn't new, and through the years, the questions have been the same: what is documentation quality? How do we measure it? And most importantly, how do we achieve it?

U.S. Supreme Court Justice Potter Stewart famously refused to offer a legal definition of pornography, saying only, "I know it when I see it." The concept of quality seems equally elusive. The literature is full of different (and sometimes contradictory) definitions, and some have even declared that a universal definition of quality is impossible.

What's the problem? We recognize quality all around us. We desire it in our products and services as consumers, and most people consider higher quality something worth paying extra for. But when we are asked (usually by an engineering manager) what quality means in our own field, and particularly when we are asked to meet a specific quality standard, we reflexively balk and claim that writing is art and an ineffable mixture of taste and style.

Attributes of Quality

In search of documentation quality attributes, our instinct has always been to ask our customers what quality means to them. Over the years, practitioners and academics have administered numerous surveys. This is good and necessary work, because products and modes of communication change over time. Each attempt to define or apply documentation quality, and each new survey, yields a fresh set of attributes. Yet each survey is only a snapshot in time of a limited set of customers about a limited set of products from a limited set of vendors. We would like to combine the results, but survey methodologies, and the questions asked in each one, change over time as well.

With enough answers, could we determine exactly what attributes bring quality? The problem is establishing which attributes *cause* quality and which are just *associated* with quality. Consider this attribute: "Every statement in the document should be necessary and correct." It's safe to say that correctness causes good quality, while incorrectness causes poor quality. Now, consider this one: "Timestamps in examples should match the product release date." This attribute is associated with good quality, but it's not causal. Examples can have old timestamps and still be correct. But they raise doubts in the reader's mind, because outdated examples tend to be inaccurate. At one time, a popular

Shutterstock/Kachka

working definition of documentation quality centered on four key attributes: correct, complete, clear, and concise. This definition sounds good; it's hard to argue that an incorrect, unclear, or prolix work is of high quality. However, completeness hasn't aged as well, because the principle of minimalism suggests it's not actually desirable. Another once-popular attribute, index thoroughness, has fallen into disuse, because users today overwhelmingly prefer search engines to indexes. Meanwhile, accessibility has become important. All quality attributes are subject to similar interpretation and evolution.

Also, quality consists of both things present and things absent: the presence of positive attributes (though some are implicit) that customers ask for, and the absence of negative attributes, which customers won't mention but don't want. Just the other day, I ordered a takeout salad that arrived with a beetle crawling through it. Previously I would not have defined quality in salads as "having no bugs"—I've never ordered a salad "hold the bugs"—and maybe someday people will regard them not as bugs but as features. For now, I'll at least make it a point of inspection.

We shouldn't just discard potential attributes; they all reflect some underlying truth. Instead, I think we should categorize them (grouping attributes into categories that one hopes remain stable over time), and then determine which ones are most important. To make sense of them all, we need a framework to think about quality, one into which every quality attribute and category can fit.

Stakeholders in Quality

I think the best way to classify quality attributes is to consider the perspective of stakeholders, a business concept familiar to most of us. Who cares if the work we do is of high quality? Our customers, obviously, but it's more than just them. Technical communicators exist in the realm of work for hire intended for an audience. The full set of stakeholders are *customers* (our audience), *clients* (those who hire us), and *communicators* (we ourselves). Each stakeholder has a different perspective on documentation quality, and all of the perspectives are valid (if perhaps not equal).

Customers

Most quality attributes are customer facing. Customers say they want clear and accurate information. Some companies define "quality" wholly as customer satisfaction, and many academic and practitioner surveys focus on what users think of information products. You'll find plenty of examples, so I won't try to add to this expanding body of knowledge, but customer quality attributes are typically grouped into these categories:

- **Audience:** The document is appropriate for the intended audience.
- **Writing:** The information is correct and clear.
- **Editing:** There are no spelling or grammatical errors.
- **Illustration:** Diagrams are crisp and clear, and screenshots are useful and legible.
- **Organization:** Topics are logically grouped.
- **Navigation:** It's easy to find and get to information.
- **Production:** Physical documents are well printed and bound; online documents are well laid out on all display devices.

Clients

Customer-facing attributes are well known, but few of them touch on the needs of clients. What do clients want? Geoffrey Bessin offers a novel approach: "Quality is 1) a well-defined process for 2) creating a useful product that 3) adds value for both the consumer and the manufacturer" (2004). Client quality attributes are almost entirely different from what customers look for. The motto of a discount chain that once did business in my area was "good stuff, cheap"; that's it in a nutshell. Every business wants the best product it can make for the lowest cost of production. The fundamental job description at my first employer was to produce "timely and accurate" documentation. Yes, accuracy was a shared attribute, but timeliness—documents ready on schedule to support a release—came first. A quality production process is timely, productive, efficient, repeatable, and predictable.

Years ago, I watched two colleagues document an email product with two user interfaces. A veteran writer was assigned to write the command-line interface manual, while a junior writer was tasked with writing the forms-based interface manual. Development was ongoing and some functions were volatile. The newbie energetically tackled the changing functions first. She met daily with developers, tracked every change closely, and regularly sent out drafts to review. Meanwhile, the veteran worked steadily, starting with the stable functions and leaving the unstable ones until they settled down near the end. When the drafts came due, the veteran was done, but the newbie had finished just the one chapter. To save the release, the rest of us had to pitch in. We got it done, but only after nights and weekends of heroic work. Judging from their reception, the resulting books were equally accurate and effective, so to *customers* they were of equal quality. But from the *client's* perspective, the work of the veteran—completed on time, on budget, and without draining additional resources—was of much higher quality than the work of the inexperienced writer, which was a debacle.

Communicators

As technical communicators, our own views on quality matter, too. Much of what we do is invisible, or implicit, in that we are paid to avoid—or at least to root out—errors of omission and commission. Readers give no credit for lack of errors but are quick to complain if they spot any. (A 2019 study by Website Planet found that Web visitors are nearly twice as likely to bounce off a site if the first page they see has a spelling or grammatical error.) Our professional standards and ethics drive us away from negative quality attributes, such as errors and typos, and toward positive attributes, such as clarity, concision, and consistency, which lie entirely within the domain of writers, graphic artists, and editors, or sometimes one person assuming all of these roles. A roomful of reviewers can make something accurate, but rarely can they make it clear or concise. That's up to us.

Doing an excellent job is great, but can you repeat the results? An individual piece of technical communication can be unique to one release of one product. Take a step back, or look over time, and you recognize the similarities between document versions, documents of the same type for different products, and types of information. (This is the theory behind DITA.) Technical communication is, in many ways, a manufacturing process, so the principles of process quality apply. The hallmark of professionalism is consistency of results; even lone writers create style guides. Getting the job done right every time requires focusing energy on elements that differ, scheduling work realistically, and obtaining regular technical reviews. These are all elements of process quality that clients might not value, but we should. The best way for us to synthesize our processes is with checklists, which are themselves collections of quality attributes.

Taking the Measure of Quality

Product managers tell us that what can't be measured can't be managed. It's difficult to demonstrate, improve, or even maintain quality unless you can measure what you're doing.

It's hard to measure poorly defined attributes. Even a simple metric like errors per page requires common understanding of what constitutes an error and a page, as well as how to take the measurement. Do we measure the entire document or just a sample? Do we measure graphics? Other attributes are even more challenging: how do you measure clarity? Can there be too many illustrations, or too few?

With careful definitions, though, it's possible to define and use documentation quality metrics. This is the theory behind publication competitions. Measurement requires an agreed-upon formula, an understanding of the range and domain, and a protocol (how to measure). Effective metrics are repeatable and objective: the number you get today must be the same if you measure again tomorrow, and my measurement must match yours. Effective metrics can be collected and combined.

You can apply some metrics to finished products (for example, is the steak cooked medium rare?) and others during the production process (did the center reach 60°C/140°F?). To apply this classification to documentation, *evaluative* metrics assess the quality of a completed information product using customer-facing attributes, which enables quality control, while *predictive* metrics assess the quality of a draft information product using client-facing attributes, which enables quality assurance. It's impractical to ask technical communicators to spend a lot of time collecting predictive metrics while they're working, so the best metrics are both valuable and easily obtained (or well worth the effort to collect). Perhaps the most readily available, crudest, and fastest predictive tool is the automated readability checker in Microsoft Word. More thorough and nearly as fast is the ISO Schematron, which flags defects in XML document source files.

For assessing customer-facing documentation quality attributes, a checklist such as the one compiled in *Developing Quality Technical Information* (Carey et al. 2014) is an effective tool.

Client quality attributes—business metrics—lend themselves more readily to precise definition and measurement, so you will more often see the most important few process metrics on a dashboard. Dr. JoAnn Hackos's *Information Development: Managing Your Documentation Projects, Portfolio, and People* (2006) is a good source for client-facing attributes.

Collecting data is good; extracting information from data is better. *Composite* metrics, measuring two or more elements at once (such as words per topic, which is information), are better and more meaningful than *simple* metrics (such as word counts, which is data).

Considering All Perspectives

Within this framework, then, quality attributes come from three sources.

- What satisfies the **customer** is the most important part of the quality equation. They are the primary stakeholders, because if they don't buy what our clients are selling, our clients will go out of business.
- The next most important source of attributes (and the most fertile source of metrics) is process, the perspective from which our **clients** view our work. Clients are the secondary stakeholder, because if our clients don't like our work, *we'll* go out of business.
- The last part of the equation is the value that we as technical **communicators** add to our own work through style guides, checklists, and personal skill. When push comes to shove, we must accede to both customer demands and client standards.

Not all quality attributes are equally valued by all stakeholders. Every company claims its customers are paramount, but tension exists between the products and services a company offers and the money that they're willing (and able) to invest in manufacturing them. For example, topic reuse and sharing—the strength of DITA—reduce client costs by eliminating nearly identical blocks of text (software developers will recognize them as "clones"). But writers in DITA shops know that customers see text optimized for reuse as vague.

A professional informed by best practices can quickly and efficiently produce high-quality results. No customer has ever complained that a document was too well written or that its illustrations were too attractive! Yet there's also tension between our urge to craft and polish prose, which we know can always be clearer and more concise, and the schedule and budget constraints of our clients. From their perspective, it's possible for us to add *too much* quality by taking too long. Our job is fundamentally a compromise: To do the best we can with the time and resources at hand.

If you draw a Venn diagram (see Figure 1) grouping customer, client, and communicator quality attributes, I believe the customer circle will be the largest, and ours the smallest. There will be attributes that matter to each

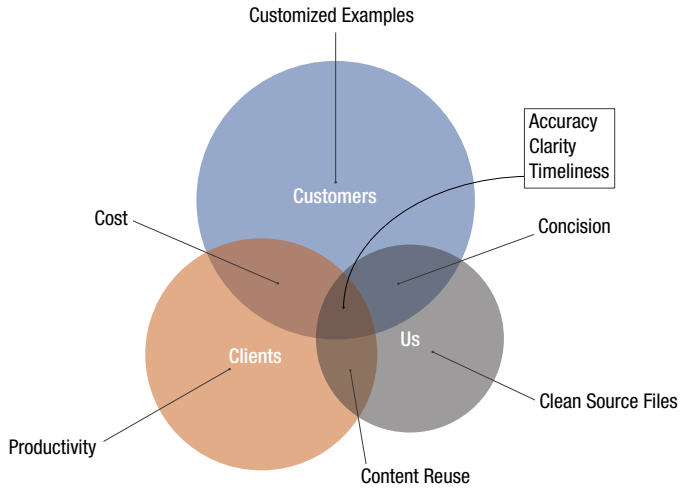

Figure 1. Common Attributes within the Quality Framework

stakeholder but not as much to the others, attributes that appeal to two stakeholder groups, and attributes valued by all three. I can't suggest how much the circles overlap, but the sweet spot will contain attributes that all stakeholders agree are important. We should focus on well-defined attributes that all stakeholders value, starting with accuracy. Where an attribute is valued by some, but not all, stakeholders, we should favor attributes valued by at least two of them. In that way, we can filter the universe of potential attributes into a manageable—and measurable—set.

This framework also gives us a test for evaluating new attributes. Do you have evidence that a potential attribute is valued by customers, clients, communicators, some combination thereof, or all stakeholders? The best new attributes are well defined, evidently valuable to all stakeholders, and easily measurable (or well worth the effort to measure).

Summary

The bottom line is that while no number of attributes can fully capture documentation quality, some attributes are more revealing than others. By adopting a framework of quality as important to customers, clients, and communicators—in that order—and by considering both product and process quality, we can classify quality attributes, determine which are most relevant and valuable, evaluate potential new ones, and focus on the most important ones—perhaps few enough to fit on a dashboard—without getting bogged down in details. i

STEVEN JONG (stevefjong@comcast.net) *has been a member of STC for more than 35 years. He contributed the "Musing on Metrics" column for the Quality SIG from 1996 to 2005. Steve is a Fellow and the recipient of the 2012 President's Award. He has served on the STC Board of Directors and the first Certification Commission, and he is in his third term as President of the New England Chapter.*

RESOURCES

Bessin, Geoffrey. *The Business Value of Software Quality.* 15 June 2004. *https://www.ibm.com/developerworks/rational/library/4995.html.*

Carey, Michelle, Moira McFadden Lanyi, Deirdre Longo, Eric Radzinski, Shannon Rouiller, and Elizabeth Wilde. *Developing Quality Technical Information: A Handbook for Writers and Editors.* IBM Press: Upper Saddle River, NJ, 2014.

Hackos, Joanne T. *Information Development: Managing Your Documentation Projects, Portfolio, and People.* Wiley Publishing: Indianapolis, IN, 2006.

Jong, Steven. "Quality Programs: Six Sigma." *STC DocQment* 9.2 (2002).

Stieglitz, Sarah. (2019). "Your Typo is Costing You 12% Extra on Your Google Ads Spend." *Website Planet.* 6 August 2019. *https://www.websiteplanet.com/blog/grammar-report/.*

Let Your Users Teach You: Document Quality and Usability Testing

Shutterstock/fizkes

By LISA D. GAY

WE KNOW that understanding a document's audience affects countless decisions: structure, order, what to include (and what to exclude), tone, and so on. One way we can approach document quality is by asking, "How well does it meet users' needs?" To answer this, we are often limited to second-hand information about users from colleagues who interact directly with customers, such as those from product management, marketing, sales, support, and others.

However, your colleagues are immersed in your company's products full time. Tasks that were confusing when they first encountered the product are now part of their knowledge about how things should work. They might even forget that that they were ever confused. That's part of building product expertise.

But that also means that your main source of information about your audience has a lot of foundational knowledge that your audience does not. Inaccurate assumptions lead to documentation that doesn't align well with your readers' needs—a mark of poor quality.

As a technical writer who has partnered with user experience (UX) designers and learned about UX design practices through that collaboration, I have discovered that usability testing is one methodology from UX design that has helped me to improve document quality by giving me a new understanding of my audience.

Real User Knowledge Makes a Difference

The first time I documented a product while following UX design principles from the start, I was amazed at the difference. The documentation was easier to write, because the features were fully intuitive. The documentation was also easier to use. I could eliminate entire sections that explained counterintuitive concepts or described how to find settings in a series of haphazard dialog boxes.

But more importantly, I could write better documentation, because I had a better understanding of our customers. Until then, I *thought* I knew my audience. I had a notion of the kind of job they had, what problem our product solved for them, and the specialized terminology of their industry. From that, I determined their priorities, what product features were most important to them, where in the workflow they might struggle, and how they understood the role the product played in their day-to-day life.

One technique in particular, however, revealed that I was making many incorrect assumptions without even realizing it: usability testing. Watching real people use the product and its documentation gave me solid information I could use to create higher-quality documents.

What Is Usability Testing?

Usability testing is easiest to understand in the context of user-centered design (UCD), a product development process that focuses on identifying and understanding users' needs. Briefly, the UCD process involves learning about your users through interviews, creating an initial product design, testing the design on real users, and then using the results of those tests to iterate on the design until it's ready to hand off to developers for implementation.

Typically, usability testing comes into play when you test the product design on people who are similar to the expected end users for your product. A technical writer observing product usability testing can learn many things relevant to improving documentation, like context, workflows, and terminology.

If there isn't a UX team running product usability testing, a technical writer can run testing on the documentation directly. Ideally, you would have access to current users or sales prospects, a person available to do note taking, half a dozen colleagues from different departments to be observers and to help analyze the results, a couple of weeks to run five or six test sessions, and a development sprint to make improvements to the product itself.

Most of us don't have all of those resources available, but that doesn't mean usability testing isn't possible. You just need to make a few compromises.

Testing Document Usability

Let's take a look at what a formal documentation usability session looks like. In each session, a single test subject attempts to complete a small number of specific tasks with the product using the documentation to help them. Observers write down observations about the user's *actions* and *words*, not attributing any meaning to them. For example, the observation "Alice re-read the second sentence very slowly" is more useful than "Alice was confused."

Usability testing also requires a facilitator, someone who is familiar with the principles of usability testing and can keep the session on track. Skilled facilitators need to juggle a lot of functions. For example, they must:

- Keep the test subject at ease. "We're testing the documentation, not you. You can't do anything wrong."
- Encourage the test subject to talk about their thinking process out loud. "What did you expect to happen when you clicked that?" "What do you think that term means?"
- Resist the temptation to help the test subject when they get stuck. "If you didn't have any technical support available to you, what would you do to figure this out?" "Good question! What do *you* think that button does?"
- Think on their feet when the test subject says or does something completely unexpected. "Could you tell me more about why you went to that page of the help?"
- Ask follow-up questions after the subjects have completed the planned tasks. "What did you find most frustrating?" "What questions did the documentation not answer for you?" "If you could rename that button, what would you call it?"

Usually, the number of new observations tapers off after about five test subjects, at which point you can analyze the results to find what patterns emerge. A good analysis occurs in two phases, kept carefully separate.

- First, list the *observations*. It's important to not interpret them, yet, so that you can avoid leaping to conclusions shaped by your assumptions. Note whether each observation was unique to a single user or occurred in several sessions.
- Second, shift your attention to *interpretation*. Discuss what each behavior means and what you might want to do about it.

These results will help you to adjust the documentation so that you can test again with the next revision.

This summary only describes the basics. You can learn more about how to plan and carry out usability testing from the resources listed at the end of this article.

It's tempting to think that other testing methods can take the place of usability testing. When you complete a document, you probably test it against the product. If your organization's release process calls for it, a QA tester might validate the procedures you wrote. These approaches are helpful and can find some kinds of flaws, but usability testing has two key differences: you're observing, not participating; and the tester is someone who wasn't involved in building the product.

These two factors might seem simple, but they explain why usability testing can make such a big impact on product and document quality. Letting real users' behaviors guide documentation decisions is the most direct way I've found to create documents that meet my audience's needs.

Making Do with What You Have

The formal usability testing process might sound like more than you can manage, especially if you're strapped for resources. Or, you might have trouble getting permission to interact with customers or show a draft outside the company. You can take a lighter approach, drawing your pool of test subjects from your colleagues.

Reach out to people in different roles—product management, engineering, customer support, marketing, and sales all have useful perspectives. Most importantly, seek people who are not familiar with the product or, at the very least, haven't attempted the specific workflow you are testing. If you have trouble finding people willing to spend time on this, seek anyone who has voiced concerns

about users struggling with the product. Position yourself as their ally in improving the customer experience. And after testing, ask if they want to recommend any of their colleagues as test subjects or observers.

What You Might Learn

The initial intent of usability testing is to improve products. When my colleagues observe testing sessions (or are participants themselves), I often hear comments like the following.

> "I never realized how complicated that configuration process is!"

> "I thought everyone understood what 'key' meant in this context, but most of the users assumed a completely unrelated meaning."

> "I thought our users needed all of the report data to decide what to do. I'm surprised they didn't want to analyze all of those metrics. How can we simplify the report view?"

This can inspire product managers and others to push for design changes that make the product easier to use—and to document. But usability testing can also help technical writers with their work. Some gems you can mine immediately:

- Finding a better word to describe a difficult, abstract concept.
- Identifying the most confusing tasks so that you can prioritize your work efficiently.
- Learning a better way to organize content in the online help.

For example, I created a getting started guide for a cloud delivery network product. Many steps applied only if the user first enabled HTTPS delivery. I prefaced each of those steps with "For HTTPS sites only." One thing I wanted to learn from usability testing was whether that wording was clear. But the test subjects showed me that I was asking the wrong question: "I don't want to have to think about what protocol I'm using, over and over through the whole process. Why not have two versions of the guide? That way I can see only the steps I need to see." The next edition of that document had two versions: one for HTTP-only sites and one for sites with HTTPS delivery. For our users, a documentation experience that only asked them whether they used HTTPS once (when picking which process to follow) was better than a single, comprehensive process that covered both cases.

You can also get valuable insights about the context in which your users interact with your product. Consider the delicate balance of password requirements. At first glance, it might seem very secure to require passwords to contain at least 15 characters of four types and to force a password change every month. But usability testing would probably reveal that users won't even try to memorize a password that complex. They'll write it down somewhere, creating a security vulnerability.

All of this information can help writers improve document quality.

Conclusion

In my experiences with usability testing, one thing has become clear: there is no substitute for watching users interact with the product. It's hard to overstate the value of observing someone try to complete a task when they don't have background knowledge of the product or preconceived notions of what the workflow should be. It's also very hard to communicate what it's like to have a user show you an entirely new understanding of a feature unless you have had the experience yourself.

I encourage you to try usability testing on your own products and documents. If you have a UX design team available, team up with them. If not, plan a testing session with the resources you have available. The books in the resources section below can guide you. Steve Krug's book, *Don't Make Me Think Revised: A Common Sense Approach to Web and Mobile Usability,* has a good chapter about conducting fast, lightweight usability testing (see the chapter "Usability Testing on 10 Cents a Day"). His website has additional resources, including a sample script, tips for observers, and a video of a usability testing session. David Platt's book, *The Joy of UX: User Experience and Interaction Design for Developers,* provides more detailed instructions and is especially helpful for navigating the process in relation to software products. Don Norman's book, *The Design of Everyday Things,* is a foundational text in UX design, with more focus on physical objects than the other two.

Do some usability testing, and really get to know your users. Your documents will be better for it.

LISA GAY, M.A. (lisagay@alumni.uchicago.edu), *came to technical writing in 2006 from a background in the humanities. She has written in several fields, including smart grid technology, cloud delivery networks, and pharmaceuticals. Currently, she documents products at RightHand Robotics, Inc., a leading provider of piece-picking robots for supply-chain logistics. Known as a passionate advocate for users, she draws on user experience design principles to create effective, frictionless user assistance.*

RESOURCES

Krug, Steve. *Advanced Common Sense.* Personal website. *http://sensible.com/.*

Krug, Steve. *Don't Make Me Think, Revisited: A Common Sense Approach to Web and Mobile Usability.* New Riders: San Francisco, CA, 2014.

Norman, Don. *The Design of Everyday Things.* Basic Books: New York, NY, 2013.

Platt, David. *The Joy of UX: User Experience and Interaction Design for Developers.* Pearson Education: San Francisco, CA, 2016.

The Evolution of Quality

By ROCHELLE FISHER

OVER THE PAST 20 YEARS, the definition of "quality" has evolved. Let's take a look at what quality used to mean and what it means now.

Then

The year is 1999. I delivered my first F1 Help project. On my desk was a bug report: "Explanations for the OK and Cancel buttons are missing."

In a huff, I went to the QA engineer who wrote the report. I demanded in what I thought was a stern tone, but was probably more of a whine, "Why in the world do the users need to be told what OK and Cancel do?"

"You explained everything else," said the young man calmly. "It looks odd that those two are missing from every page."

My future-self guesses that the QA team probably had a quota of bugs to write, and this was filler. My then-self didn't know this was a thing. She thought back to 1997, to her teacher's words:

> "The goal of technical writing is *rapid information retrieval*. We accomplish that through writing according to the three Cs: Comprehensive, concise, and consistent."

As far as I knew, my writing was concise. It was consistent with my senior writer and with the style guide. If I added OK and Cancel, my document would be comprehensive.

Now

Today, I believe that the definition of quality has changed. There are different ways to define quality. Let's look at two: the ideal and the reality.

The Ideal

With the modern dispersion of technology, today no technical writer would waste time or space explaining OK or Cancel. In 2017, Jason Silva said, "A young kid in rural Africa with a smartphone has better communications technology than a head of state—than a president—had 25 years ago." Our users have more knowledge, and less patience, than they had 20 years ago. If we waste our space with knowledge that can be assumed, we lose their trust that we have something to offer them. When users see something similar to "Click OK to close the window" in the first procedure, many will decide that the complete document is too remedial and close it. They will look for their answers in Google or open support tickets. Our work is undone.

If we upgrade our ideals of quality documentation, we can fulfill the vision of helping our users with *rapid information retrieval*. We evolve our definition of quality to meet the evolved requirements of our users.

- Rather than *comprehensive*, aim for *relevant*.
- Take *concise and consistent* to a new level with *controlled*.
- Expand our goals to include accountability by definition and make *accurate* an explicit characteristic of quality results.

Shutterstock/Alexander Supertramp

Relevant

If the product is basically intuitive, you do not need comprehensive documentation for every possible way to open a window or change a light bulb. If you know your target audience well enough, you can create product deliverables that answer questions that users ask. These questions can be anything from "What can this product do for me?" to "How do I respond to this emergency?" to anything in between.

Relevant is not a replacement for comprehensive. Quality documentation explains every feature; it is comprehensive. It also takes into account how much we explain and in what order. A *comprehensive* document explains everything that the developers give us. A *relevant* document explains what the user needs to know. Relevant technical writing is delivered in a medium that is expected by the users. It might not be a document at all. It could be video tutorials, self-service articles, or chatbot answers.

Let's look at an example. In version 1.8 of a product, the admin guide mentions a new feature: Users are required to log in with Two Factor Authentication (2FA). We have all used 2FA, though not all of us know that name for it. It is when you are required to enter a code from a smartphone application, such as Google Authenticator, in a web user interface. In the 1.8 guide, the technical writer wrote more than 200 words about setting up 2FA with different vendors. Later, we realized that the admin who clicks **2FA Required** on the product does not set up the authenticator. All those words explained what 2FA is and how it works. There was nothing for the user to do. In version 2.0, we made the documentation relevant. We replaced the 200 words with, "Users will get the QR code and instructions on their next log in to the Console."

Be aware! If you give only instructions for what the user can do, your support engineers will complain. Users will open tickets to ask how something works, or what happens in the backend when an option is selected. You cannot make a hard-and-fast rule that your documentation will contain only what you think is relevant for the users *until you learn what your users think is relevant.*

Here are some tips that I use to write what is relevant.

- Use the product. If you write about using regular expressions (regex) in your product, be as knowledgeable in regex as your target audience. You will know what to document and how much. If you never use regex, you might end up hiding the relevant points in a mess of industry-standard knowledge.
- Build a relationship with your company's professional services group (sometimes called customer success or similar names—the team who works with users, after pre-sales and before customer support). If possible, shadow an on-site visit.
- Lurk in webinars and lectures. Make a note of questions that are asked, and make sure that the documentation answers them.
- Take free, online courses in the field of your target audience. Make sure you know the terms and concepts that are assumed knowledge.
- Search for "this means that" and "in other words" in your document. These are red flags that you might have wasted words explaining the technology to yourself.

Controlled

A *concise* document does not repeat information and does not use multiple words when one will do, but a document that conforms to a controlled language (CL) standard is, on average, 30% more concise than one written without a CL (Braster).

A *consistent* document might simply be consistent with a style guide, but a document written using a CL reaches a new level of consistency. The main purpose of a CL is to control the vocabulary. One word has one definition, and that definition is represented by that word only.

Even without a formal CL, you can control your language with a style guide, if it limits allowed grammatical forms and vocabulary.

Here is an example that offers *more with less* when the language is controlled.

5. In the last step of the wizard, click **Finish**.
6. When the install has been completed, restart the computer.

With a CL standard, this example becomes the following.

5. Click **Finish**. One hour is necessary to complete installation.
6. Restart the computer.

In this example, install is sometimes used as a noun and sometimes as a verb. This can cause confusion in longer sentences and for non-native English readers. The CL determines that install is a verb, and installation is a noun.

The original text tells the user when to start the action (when the last step of the wizard appears or when the installation is done), but the numbered steps indicate that the user should perform one action at a time. Why say, "When the previous action is done, do the next action," when you can give information that is meaningful? Do not use the vague "when." Tell how long the user must wait, or show the result, or remove that unnecessary text.

The best tip I can give for writing with a CL is to choose one ("Controlled Natural Language"), and then get professional training in how to use it.

Accurate

I've seen many advertisements for technical writing courses that promote technical writing as something that ex-pats can "easily do" in a non-English speaking country or as a good part-time job for retirees of any background.

I cringe when I see these. Technical writing is no longer (if it ever was) simply writing. A quality technical writer is a tech lover, a fast learner, and a person who holds himself or herself accountable for the final deliverable. We cannot rely on others to do the technical work for us. We must run CLI commands. We must write scripts that use the API we are documenting. We must be hands-on. We must ask the questions that the users ask. We must get the most accurate information possible.

Here are some tips that I use to write what is accurate.

- Help QA when possible. If your company has a bug bash event, join it.
- Learn the operating systems that your users use. For example, if you learn enough Linux, you will know to add the dot before the slash if your SME forgets it.

The Reality

Relevant, controlled, and accurate: these are the ideals to reach for. You can get training in a CL, learn your product line inside and out, and obtain an understanding of what the typical user needs. You can create quality deliverables given the time.

This brings us to the second definition of quality.

The reality of many companies is Agile sprints and frAgile releases. Quality is not an idea. It is a specific definition in the scope of a high-level work plan. Often, the definition of quality for documentation in this reality is, "it exists."

We have our benchmarks, our idea of the perfect deliverable. We constantly strive for relevant, controlled, accurate results. It is good that we do so, but given two weeks to create documentation for a new product, we must become project managers. We get the scope and determine the priorities. We make impact plans and schedules for stages of delivery. We produce what is required according to the project plan and the product owner's definition of *done*.

That is quality for *that* project. The definition of quality will change with the next sprint. Our ideal of quality guides us in our actual work, but it must never hinder us from fulfilling the definition of the project. We cannot let a release be late due to documentation. Sometimes, we must deliver a 100-word description of the product in concise, accurate language that is relevant for a specific vendor and that vendor's audience.

And those 100 words will be all that we are allowed to deliver. And we have to be okay with that.

ROCHELLE FISHER (rochellef@sentinelone.com) *is a documentation team manager. She began her career in technical communication in 1997. Rochelle loves Simplified Technical English (STE) and has trained newbies in it, though she insists that a team who is serious about writing better, faster, and cheaper get some professional CL training.*

REFERENCES

Braster, Berry. "HyperSTE – ASD STE-10 Software: Standardizing the Content of Technical Publications." *http://www.s1000d.ru/userforum/presentations/Day_2_Track2_01_Tedopres_HyperSTE.pdf.*

"Controlled Natural Language." *Wikipedia. https://en.wikipedia.org/wiki/Controlled_natural_language.*

Silva, Jason. Keynote Speech at Teradata Partners, 2017. *https://youtu.be/7xGCmGt-yTo?t=376.*

Just Because the Spelling Is OK Doesn't Mean It's Quality Content

Shutterstock/ GaudiLab

BY ALAN J. PORTER

THE DICTIONARY DEFINES the word quality as "the standard of something as measured against other things of a similar kind; the degree of excellence of something." But what does that mean in terms of content? This question made me ask, "Do we, as a profession, measure our degree of excellence?"

I recently ran a quick poll using my Twitter account (*@TheContentPool*) asking what other content professionals think of when they hear the phrase "content quality." The answers were along the lines that I expected, in that they were focused on the mechanics of producing good content.

The responses covered points such as:

- Grammatically correct
- No typos
- Consistent
- Preferred wording
- Unambiguous
- Enriched for search and discoverability
- Follows the organization's style and voice
- Enriched with links when references are made
- Has an explicit audience
- Has measurable goals
- Suitable for use in multiple contexts.

These are all great points and definitely things that we should all be thinking about as professional communicators. From a content convergence perspective, they are also goals that we should be encouraging other content producers across the organization to strive for, as well. We can help improve the quality of the content that our enterprises produce by leading by example.

That doesn't necessarily mean that we need to hold, and lead, classes on writing skills (although that can often be useful). What we *can* do is reach across functional boundaries to other content creators to ensure that we are all using the same words to mean the same things. Ideally we should have, or develop, a common vocabulary and taxonomy that can be used across the organization.

When we think about producing quality content from a holistic viewpoint, we should be aware, and make others aware, that the content could be used in different contexts and needs to be created with that in mind.

Creating content—no matter how polished it might be—just for the sake of creating content (as often happens with content produced in an organizational vacuum) is a waste of time and resources. I refer to this as "so what" content. It may be well written, but if it serves no purpose, and the reaction after consuming it is "so what," then it isn't a quality experience.

One of the respondents to my Twitter survey asked, "What does it (content quality) mean to you, Alan?"

For me, it means that content has the purpose I alluded to earlier. As some of the survey respondents suggested, good content should be written for the audience who will consume it, and have measurable goals. But I believe to be truly high-quality content, the measurable goals shouldn't just be internal ones (like click rates) but also those focused on helping the customer succeed. Did the content we produced deliver value to the customer? Did it help them with what they needed to do, answer a question, carry out a task, complete a transaction?

We can produce all the content we like that is grammatically correct, spelled correctly, lacking typos, and so on, but it's no good getting the mechanics of content production right if we don't help those who use our content.

What does "content quality" mean to me? It means delivering value.

As pre-sales content and post-sales content begin to overlap, Alan Porter provides the latest insights about our role in that evolution in Convergence Conversations. Learn through this column to build bridges and form synergies with your counterparts in marketing. Contact Alan at *ajp@4jsgroup.com* to ask a question or propose a topic for him to cover in this column.