User Documentation and Customer Care

Cem Kaner, Ph.D., J.D. & David Pels, B.A.

In recent years, the *Customer Care Survey of Service and Support Practices in the Software Industry* has consistently reported that only about half of software publishers put their documentation through a formal test. We thought that these numbers were low, so we checked them at the Software Testing, Analysis & Review (STAR) conference (Orlando, May 16, 1996). During a plenary session, Kaner asked attendees (software testers) whether their groups tested their companies' user manuals. Confirming the *Customer Care* data, at least half the room stood up to signify that their companies did *not*. This means that reputable companies are not testing their manuals -- companies who don't care about quality don't spend money to send testers to STAR.

Bad Documentation Results in More, Longer, and Tougher Support Calls

Some people don't see this as a problem because they believe the *myth* that no one reads documentation. *They are mistaken*. According to recent data from Dataquest, 85% of customers solve their own problems without calling for support. Many of these people must be using their manuals. One of us recently studied this issue for a client. (This was not a mass market software publisher. Consumers might be less likely to check their documentation.) The large majority of customers surveyed claimed that they checked their manuals before calling for support. Also, when customers did call for support, the large majority claimed that they had checked the documentation first, and most of those could tell us why the manual hadn't answered their question. The problems they cited were unsurprising. They couldn't find what they needed because:

- the index was incomplete or wrong or pointed to irrelevant information
- the table of contents provided no hint as to where to find the information
- the information was wrong, incomplete, incomprehensible, or spread across too many places in the book to be useful

Sound familiar?

If the manual and help were correct, useful, and understandable, how many support calls would never be made, and how many others could be handled much more quickly?

Errors that mislead customers about the capabilities of the program can lead to repeated, frustrating support calls and unpleasant expressions of mistrust by customers. Left uncorrected, this can lead to employee dissatisfaction and turnover.

Errors also lead to longer calls. Along with giving the customer the correct information, the support representative has to clear up the customer's confusion when this conflicts with a mistake in the manual. The support rep has to take the time to explain that there was a mistake in the manual, and apologize for the mistake, and probably has to repeat the correct information, and should then make the effort to recreate customer loyalty. All that time, wasted.

Additionally, errors can increase your company's legal liability.

Documentation Errors Can Lead to Liability

Most software sales in the United States are governed by the Uniform Commercial Code (UCC). Section 2-313(1) of the UCC states that an express warranty is created by "any affirmation of fact . . . which relates to the goods" or "any description of the goods" made by the seller to the buyer. Statements of opinion are not warranties, but factual statements (things you can prove true or false) are warranties even if the seller doesn't say "warranty" and doesn't intend to create a warranty. If the seller says it to the customer, and the customer believes it, the seller has made a warranty.

(Under the current UCC, some States would allow a seller to argue that an erroneous description should not be treated as a warranty if the buyer didn't become aware of the description until after the sale. The latest (November 1, 1996) drafts of UCC Article 2B (which will govern software licensing) and of revisions to Article 2 (sales of goods) both narrow this loophole.)

An express warranty cannot be disclaimed. If you say it, you are stuck with it. A breach of warranty is a breach of contract, for which the customer can demand a refund, and demand damages, and sue.

Software user documentation is a collection of affirmations of facts that describe the goods. It will be hard to argue in court that a manual or help system are not warranties. A company that releases its manuals without testing them is running a significant risk of creating warranties that it cannot live up to. It is handing each of its customers a written invitation to sue it.

Furthermore, even if the seller's lawyers can somehow convince the judge that the documentation is not an express warranty, the statements in the documentation will still be relevant evidence of how the program is supposed to work. Errors in the manual that create false expectations about the way the program should work will be gifts to the buyer, making it easier to win a suit for fraud, breach of contract, or deceptive or unfair trade practices.

For additional information, see C. Kaner, "Liability for defective documentation," *Software QA*, Volume 2, #3, 1995, p. 8.

You Can Improve the Documentation

In contrast with the low Testing numbers, the Customer Care Survey reports that about 80% of the industry give their documentation to the Technical Support group for review. By carefully testing this documentation, you can make a difference in its quality, and therefore lower your support burden and your company's risk of lawsuits.

We urge you to fight for the time and budget necessary to do this job well.

A thorough test of a manual takes about 15 uninterrupted minutes per page. In the best case, the same person works through the entire manual. To meet a deadline, though, you may have to assign different chapters to different people.

It's important to get the manual early enough for your comments to make a difference. Last-minute reviews can only address critical errors. You want the first circulating review draft.

How to Test the Documentation

Without clear instructions to focus on technical accuracy, many support staff focus their comments on grammatical and spelling errors. We recommend against this. Leave these for the editor. Explain to your staff that paying attention to spelling, grammar, and stylistic errors will distract them from the technical issues that no one else will catch. We recommend that your staff check the following:

- Check every written document that goes to customers. We say "the manual" but we also mean the readme file, the help file, the various collaterals, etc.
- Verify every statement of fact and every reasonable implication. *You must do this at the computer, actually checking the program to make sure that the statements are correct.*
- Check the index. Does it lead you to answers to the questions your customers will most often ask? Are the page numbers in the index correct? You may have to do this on a later draft than the main review draft, but do it as soon as you can and as well as you can.
- Audit the completeness of the manual. Does it cover all the features?
- Check the locations of the files. Does the book tell the customer to look for a file in the wrong disk or directory? If it mentions a file, is the file name still correct?
- Check the accuracy of all mail addresses and phone numbers. For any special offers, make sure that the customer is given that offer's correct phone number. Check the scope of the numbers too. If an 800 number is only valid in one state or country, check that the manual says so and provides a different number for other callers. Check the e-mail addresses, the support hours, and everything else that provides contact information or sets service expectations.

You Can Add Further Value to the Documentation

Along with checking the accuracy of what has been written, you will be disappointed by what has *not* been written. Three sections of many manuals are incomplete or missing:

- Troubleshooting advice and workarounds.

- Error messages (every message that the program can produce should be listed and explained).
- User tips for more easily or more efficiently doing common tasks.

These sections are harder to research and write than the main body of the manual. According to Joann Hackos (see her excellent book, *Managing Your Documentation Projects*, Wiley, 1994) the average manual takes about 4 to 8 hours per page to develop. Troubleshooting material takes much longer, often averaging over 20 hours per page. (This number is based on Kaner's experience managing doc groups and product development teams.) Writers find it difficult to develop this material because they don't understand what is needed. They don't know what problems to discuss or what information to provide that will be helpful.

There is strong customer care value in making sure that this material exists, that it is accurate, and that it provides usable, useful information. It will often be the case that the only people who can provide this material work in your group. We recommend that you negotiate an agreement with the writers to provide first draft troubleshooting material, which they will edit and improve. When you don't have the original information (such as error messages), we recommend that you consider co-developing first drafts of this material with the testing group.

For a more extended discussion of the process of documentation testing, see Chapter 10 (Testing Documentation) of Kaner, Falk, and Nguyen's book, *Testing Computer Software* (2nd Edition, International Thomson Press, 1993).

==========

Sidebar (long)

Documentation Errors and Liability

Most software sales are governed by the Uniform Commercial Code, which states that an express warranty is created by "any affirmation of fact . . . which relates to the goods" or "any description of the goods" made by the seller to the buyer. Statements of opinion are not warranties, but factual statements (things you can prove true or false) are warranties even if the seller doesn't say "warranty" and doesn't intend to create a warranty.

An express warranty cannot be disclaimed. If you say it, you are stuck with it. A breach of warranty is a breach of contract, for which the customer can demand a refund, and demand damages, and sue.

Software user documentation is a collection of affirmations of facts that describe the goods. It will be hard to argue in court that a manual or help system are not warranties. A company that releases its manuals without testing them is running a significant risk of creating warranties that it cannot live up to. It is handing each of its customers a written invitation to sue it.

Furthermore, even if the seller's lawyers can somehow convince the judge that the documentation is not an express warranty, the statements in the documentation will still be relevant evidence of how the program is supposed to work. Errors in the manual that create false expectations about the way the program should work will be gifts to the buyer, making it easier to win a suit for fraud, breach of contract, or deceptive or unfair trade practices.

Sidebar (shorter)

Under the Uniform Commercial Code, an express warranty is created by "any affirmation of fact . . . which relates to the goods" or "any description of the goods" made by the seller to the buyer. Statements of opinion are not warranties, but factual statements (things you can prove true or false) are warranties even if the seller doesn't say "warranty" and doesn't intend to create a warranty.

An express warranty cannot be disclaimed. If you say it, you are stuck with it. A breach of warranty is a breach of contract, for which the customer can demand a refund, and demand damages, and sue.

It will be hard to argue in court that a manual or help system are not warranties. A company that releases its manuals without testing them is running a significant risk of creating warranties that it cannot live up to. It is handing each of its customers a written invitation to sue it.

==========

Cem Kaner, J.D., Ph.D., is Professor of Computer Sciences at the Florida Institute of Technology.

Prior to joining Florida Tech, Dr. Kaner worked in Silicon Valley for 17 years, doing and managing programming, user interface design, testing, and user documentation. He is the senior author (with Jack Falk and Hung Quoc Nguyen) of TESTING COMPUTER SOFTWARE (2nd Edition) and (with David Pels) of BAD SOFTWARE: WHAT TO DO WHEN SOFTWARE FAILS. He is the co-founder and co-host of the Los Altos Workshop on Software Testing, the Software Test Managers' RoundTable, and the Workshop on Heuristic & Exploratory Techniques.

Dr. Kaner is also an attorney whose practice is focused on the law of software quality. He is active (as an advocate for customers, authors, and small development shops) in several legislative drafting efforts involving software licensing, software quality regulation, and electronic commerce.

Dr. Kaner holds a B.A. in Arts & Sciences (Math, Philosophy), a Ph.D. in Experimental Psychology (Human Perception & Performance: Psychophysics), and a J.D. (law degree). He is Certified in Quality Engineering by the American Society for Quality.

David Pels started in computing at the retail level, selling computers, electronics and software. He then managed customer care in retail for 4 years and has been managing customer care for software and hardware publishers since 1986. He currently manages Product Support and Customer Services at Snap-on Diagnostics. Pels holds a Bachelor of Arts degree in Geography, with extensive continuing training and development in customer care, human resources, and training. The views expressed in this article are those of David Pels and not necessarily of Snap-on Diagnostics. Pels is an Eagle Scout who has been an Adult Education instructor for the Boy Scouts of America, including Wood Badge and High Adventure training.