Making the Manual Work: Techniques for Validating Computer Documentation

by Neal Margolis

Scene: The Conference Room of Flashex Software Corporation in Silicon Valley. Post mortem for Account-a-Calc, a product that shipped 3 months ago. Around the table, sipping soft drinks and munching popcorn, are the main players on the product development team. The leader from Customer Support is stating his case.

Harry Briggs (Customer Support): What the field seems to be telling us is that there's a lot of trouble with the documentation package. Tech Support reports that the hot line...

Joe Fisher (User Publications): Now wait a minute. We worked for 3 weeks, picking that package apart and putting it back together until the writing group couldn't recognize their own design.

Harry: Hey, I know what we all went through, Joe. That's not what I'm talking about. The fact is that the hot line reports a humongous amount of time spent telling callers where to look in the Reference Guide. Not only that, but...

Joe: But the thing has an index! We all said it needed an index, remember?

Harry: Hmm. I guess you're right, Joe, but how do you explain...

Joe: And tabs! Red ones, so they stand out. And lower-level bleed tabs, and an easel built into the Tutorial binding so it stands up, and...

Harry: And four-color printing on the covers, and three editorial passes, and two technical reviews, and we know all that, dammit! What I'm trying to tell you is that *the documentation package doesn't work!*

Joe (starting to whimper): But it was apparent to everybody that ...

That's right, Joe, you did all the right things. But a package that works *apparently* may not work in the real world—may not be valid *empirically.*

All the tabs, indexes, editorial passes, two-color printing, and tutorials don't prove that the documentation works with real-world users. For empirical validity, you need to give the manual a test that approximates its use in the real world. You need to measure the output of the manual to prove that it's "up to spec." If it's not, you revise it. You *make* it work.

Making it work through validation is common in the training community, where empirical development techniques were first developed. The high-tech community, though, is not used to looking at the manual as an instructional item, and techniques for testing documents are not readily practiced. The computer industry is slow to see the user as a learner.

Validation of computer manuals, like validation of instructional manuals, makes the user/student a court of last resort and, thus, tends to cut down on arguments about what will or will not work. Without data, it is easy to bicker back and forth about whether a manual needs tabs. However, when you see a user waste 10 minutes flipping pages to find out how to save a file, the need is clear.

Validation allows the writer to write *lean*—a discipline that calls for the bare essentials. Validation points out where added detail is needed much more readily than where detail needs to be cut.

Finally, manuals get completed sooner because controversial material can be included with the qualification that the material *prove* itself in a trial situation.

Validating a manual, then, enhances the quality of the manual as it makes the development process more efficient. The result is that the user (and the writer) gets a better deal.

Issues in Validating Documentation

To validate a manual means to test it in order to determine how well it supports the user and where it falls short, and then to improve it. While a manual might possess a degree of *apparent* validity, we refer to validation as an *empirical* process—we select representative *test users* and watch them working with the manual and product to perform realistic tasks. If the test users perform the tasks at a stated level of proficiency, then the manual is valid.

For example, we can measure a manual's *accessibility* by giving the user a reference task (say, to find the value of the default bottom margin) and then seeing how long it takes the user to find the answer in the manual. If the user performance does not meet or exceed the standard—if the user takes too long, or fails to find the answer—we need to fix the manual.

To make sure the manual responds to an appropriate audience, we try it out with *test users* with different backgrounds, experience, and uses for the product.

Validation techniques like these, while not widely appreciated in the computer industry, are considered a subset of *usability testing*. The rest of this section will take a closer look at the main issues in validating user documentation.

Figure 1.
Sample Accessibility Items

These sample items test the accessibility of the documentation. A test of search efficiency can be made by timing the user's response — the faster the user determines the correct answer, the greater the document's search efficiency. Additional test items can be developed to test the actual acquisition of concepts as well as the ability of the user to perform critical tasks.

Write the name of the book and the page number in the book on which you can find the answers to the following questions:

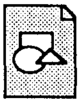
a. How do you change the number format library?
Book: ______________________ Page: _____

b. How do you add a specific number of rows into a table?
Book: ______________________ Page: _____

c. How do you loosen kerning?
Book: ______________________ Page: _____

d. What is "flow tolerance"?
Book: ______________________ Page: _____

e. What is a "Library Collection"?
Book: ______________________ Page: _____

f. What does this icon represent? 

Book: ______________________ Page: _____

g. How do you add a pattern set?
Book: ______________________ Page: _____

h. What is "Layout"?
Book: ______________________ Page: _____

Documentation Goals

We measure documentation validity against two general *goals:* a tutorial goal and a reference goal.

The tutorial goal is the need to familiarize the user with basic aspects of the product, such as terminology, concepts, and procedures. The tutorial goal may also suggest the need to teach the structure of the documentation package itself. We measure tutorial validity by:

- observing test users as they work through tutorial sections of the manual
- asking test users questions to determine whether they learned the critical material
- asking test users to perform basic tasks and observing how well the tasks are completed

The reference goal is the need to enable the user to find specific information quickly. We measure reference validity by giving the test user a question and seeing how effectively he or she finds the answer in the manual.

Validation Test Items

Validation *test items* are the actual questions and tasks that are put to the test user in order to determine if the goals are met. Taken together, the test items are the manual's performance test.

In developing test items, we ask, "What tasks will we require the test user to perform that will indicate that the manual is doing its job?" Test items can be grouped into five categories:

Accessibility. Accessibility items measure how effectively and efficiently a user can find information in a manual. Refer to Figure 1 for sample accessibility items.

Knowledge. Knowledge items test the ability of the manual to teach recall of concepts and procedures. From an instructional point of view, the fact that a user can show knowledge of a concept indicates (but does not prove empirically) that he or she can use the concept productively. Refer to Figure 2 for sample knowledge items.

Performance. Performance items test the ability of the product and the manual to teach the user to do important tasks. A problem with performance items is the difficulty of separating the effect of the product from the effect of the manual. Refer to Figure 3 for sample performance items.

Tutorial walk-through. In this type of test, the user is asked to work through a guided tutorial while the validator looks on to detect unclear or unreasonable instructions. Refer to Figure 4 for an example of a tutorial test script.

Attitude. Questions about the user's attitude are frequently useful in disclosing motivational problems that other tests cannot get at. In debriefing sessions with users testing an online HyperCard presentation, Barbara Harvie of Software Publishing Corpor-

Figure 2.
Sample Knowledge Items

These sample items test the ability of the documentation to teach recall of concepts and procedures. From an instructional point of view, the fact that a user can show knowledge of a concept indicates (but does not prove empirically) that he or she can use the concept. The test may allow the user to use the product and the documentation to get the answers, or to use only the document.

Answer the following questions:

a. What is the procedure for changing the number format library?

____

____

____

b. What is the procedure for entering a specific number of rows into a table?

____

____

____

c. What is the procedure for loosening kerning?

____

____

____

d. What is "flow tolerance"?

____

____

____

Figure 3.
Sample Performance Item

These sample items test the ability of the product and the documentation to teach the user to perform important tasks. A problem with this type of test is the difficulty of separating the effect of the product organization from the effect of the documentation.

Perform the following tasks. The facilitator will help you if you run into trouble.

a. In the sample provided, change the number format library from ______ to ______.
b. Enter 4 rows in to the sample table provided.
c. In the sample provided, loosen the kerning form ______ to ______.
d. Add a pattern set.

ation asks these questions (among others):

- Did you feel stupid or frustrated at any point?
- Do you feel that you learned anything?
- Were you surprised by anything? Where?
  Did you have fun?
- Did the organization seem clear and logical?
- Were you bored at any point? Where?
- How do you feel about the level of information? Too much? Too little? Not relevant? Useless?
- How do you feel about the overall metaphor? Was it clear? Did it get in the way? Did it contribute to the overall enjoyment?

Failure on a particular test item by several users indicates that the documentation needs to be revised or supplemented.

Where do the test items come from? They are derived by analyzing the product, analyzing customer specifications, and interviewing product technical and marketing experts to determine what is expected of the product and of the user's experience with it.

Figure 4.
Sample Tutorial Test Script

This script shows how a tutorial test might be introduced to a test subject who represents a segment of the user audience.

Hello, and welcome to the test of the *(Product)* documentation.

Today, you'll help us determine whether the *(Product)* tutorial documentation reaches it's main objectives: to get you started with *(Product)*, to familiarize you with its main elements and how they work together.

Your part in the test is to put yourself in the shoes of a new *(Product)* user. You have decided to work through the tutorial to get a general picture of what the product can do and how it works before you plunge in and start to use it.

Work through the tutorial following all instructions. If you see anything that's unclear, or incorrect, or just plain silly, make a note of it and we'll discuss it later.

I'm here to make sure you work smoothly and efficiently by helping you over trouble spots. So, if you have a problem that keeps you from proceeding, let me know and I'll help you out.

After you've completed the tutorial, we'll talk about your impressions. Where the document was not helpful, we'll ask for your suggestions for improving it.

Keep in mind that we are not here to test you. Instead, we are testing the documentation in order to find ways of making it better. Once again, thanks for your help.

Turn to page ___ of *(name of manual)* and start the tutorial.

After this welcome and introduction, the test subject sets to work under the watchful eye of the validator who takes notes and remains on hand to help the subject over trouble spots.

Selecting Test Users

Test users should reflect the characteristics of your user population as accurately as possible. Identifying these characteristics is not a trivial affair; engineering, marketing, and user support people—all the people collaborating in bringing the product to life—need to help pin down the users in a written profile.

There are several points to consider when developing a user profile:

Computer skill level. The test users should represent the range of computer expertise expected of product users. If your product marketing is targeted to experienced programmers, for example, you ought not to focus the validation on first-time computer users. If the product is meant for a wide range of users, then you might have to select a wide range of test users as follows:

- people who use the computer for more than two applications
- people who use the computer for one or two applications
- people who use a computer, but not this kind of computer
- people who have never used a computer, but have used a keyboard
- people who have never used a keyboard

Learning style. Different users have different learning styles. Information collected from a test user with one style may not apply to users with other styles.

- *Student:* This person works through the manual in a linear fashion and adopts all its recommendations for becoming familiar with the product. He or she does not experiment.
- *Leaper:* This person immediately sets his or her own objectives

for learning about and using the product. He or she likes to experiment, and accesses the manual more-or-less randomly from the start.

Learning stage. The user moves through several stages as he or she becomes productive with the product. The user's documentation needs at the start of the learning experience are not the same as those later in the learning experience.

- The first stage is a user who is approaching the product cold, without any training or preparation.
- A later stage is a user who has had a few hours to get familiar with the structure of the product and the manual.

Professional environment. Depending on the work that the user does, expectations and documentation needs will be different.

- word processing/page layout
- graphics
- numeric analysis
- management presentations
- reports

Number of test users. Validation is an activity for gathering information—it is not an activity for demonstrating high levels of statistical confidence. Writers and editors who validate their manuals soon learn that the quantity of new information diminishes rather quickly over a certain number of trials. Of course, as the complexity of the manuals increases, and as the product's market broadens, you will want to increase your test user sample.

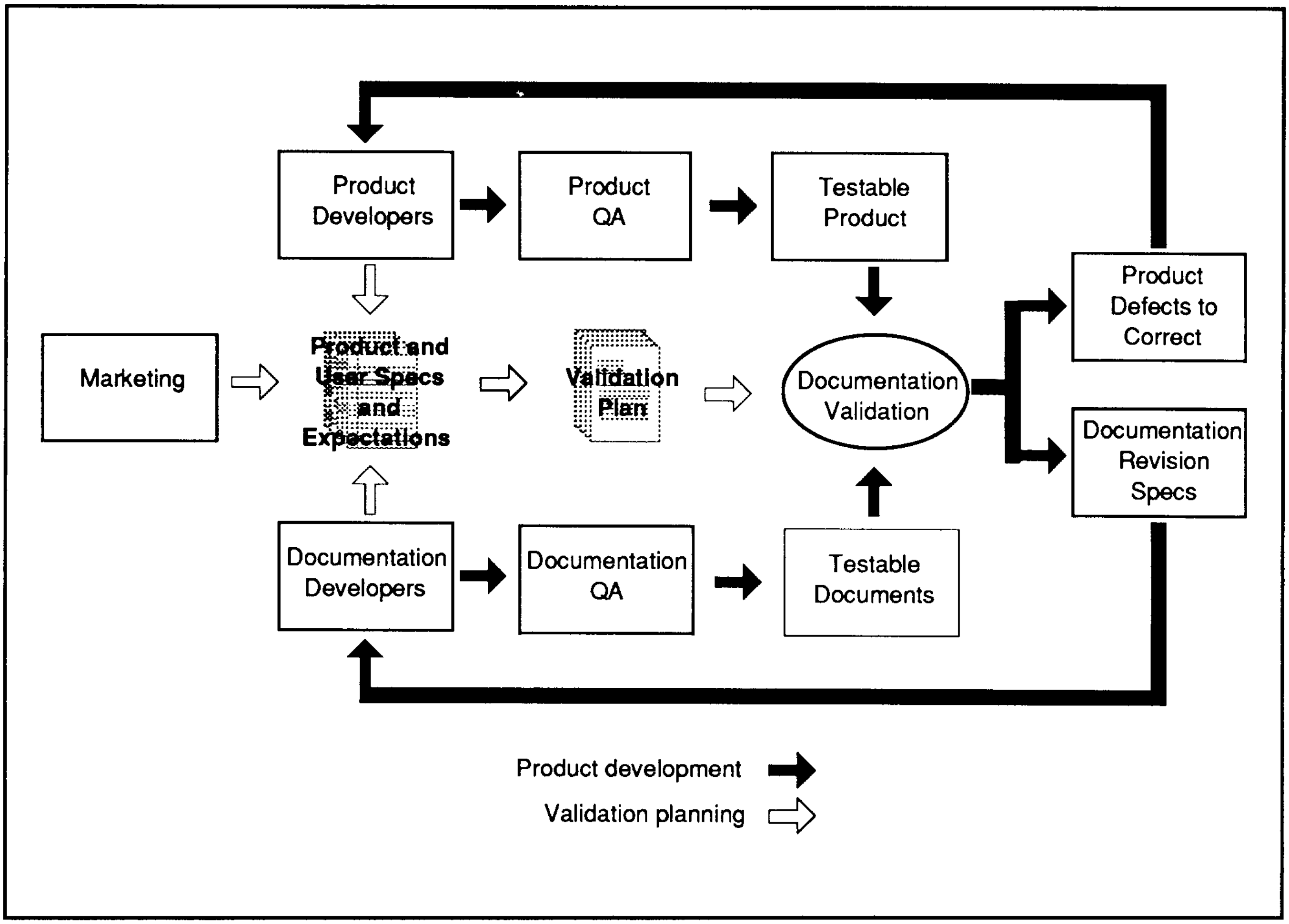
For example, you will probably find that three test users are sufficient for validating a low-end graphics product manual with a 30-page tutorial and a 70-page reference section.

Coordinating Validation

Validation is a collaborative process. It cannot work without the understanding, acceptance, and participation of the major players in the product development cycle. Figure 5 shows how the validation process might fit into a typical product development scheme.

Just as several people need to contribute to creating the validation plan, these people need to be informed of the validation results.

Figure 5.
The Validation Process



- Writers and editors need to know how well the manual performs—and where it doesn't—so they can revise it effectively.
- Marketing needs to know what can reasonably be expected of the user and what cannot.
- Software engineers, especially software test people, need to know about interface failures that are observed in the test situation so that the failures can be fixed.

Note also that while the validation process is meant to test the effectiveness of an existing documentation package, the requirements, including the validation goals, are critical inputs to the design of the package. For this reason, the validation effort needs to be coordinated so the validation plan, tests included, is completed early in the documentation design step. In this way, the writers use the validation tests as a specification against which to design the documentation.

Validation Cost

You will have to estimate costs for testing manuals and fold these costs into the product development budget. Here are some important factors that will affect your costs for validating manuals.

Number of users tested. As the number of test users grows, the amount of information to be collected and analyzed tends to grow as well. So does the cost of the time spent by the test users.

Complexity of the manual. As the number of volumes in the documentation set increases, and as the variety of presentation formats increases, so does the effort to test the set of manuals.

Extent and complexity of the test. As the test becomes exhaustive, and as more validation users are required to test all aspects of the documentation, the effort to administer the tests and to analyze the results increases.

Quality of the documentation. The closer the manual comes to meeting its objectives, the fewer the revisions that will need to be specified. This does not mean, however, that you can reduce overall costs by polishing the manual to perfection and then side-stepping the validation process. *A priori* perfection is elusive and may never be attainable. It is the validation process itself which provides the information about how far from perfection your efforts lie.

The best policy, again, is to write lean—to include in the manuals only the most relevant information and to assume that a validation test will point out missing material much more readily than surplus material.

Benefits to the Writer

A top-down management initiative is not the only support required for a motivated validation effort. At a densely networked organization like Apple Computer, Inc., where writers as well as managers provide significant direction, a bottom-up drive is necessary as well. Scottie Zimmerman of User Pubs at Apple promotes testing by publicizing benefits to the writer, as shown in Figure 6.

Validation Steps

This section describes three steps (see Figure 7) in validating user documentation: creating the validation plan, conducting the validation test, and writing revision specifications.

Creating the Validation Plan

The validation plan details the conditions under which the product will be validated as well as the particular validation procedures.

Who Creates the Plan?

The direct responsibility for writing the validation plan is best given to the designers of the documentation package. As mentioned earlier, though, other members of the product development team need to provide information for the plan. Since, for the most part, the validation test will determine whether the expectations of the documentation are met, these expectations need to be stated early in the development cycle.

Marketing people should describe the product positioning and the users for whom the product is intended.

Technical and engineering people should describe the common uses to which the product will be put.

Both marketing and technical team members should review the draft plan and comment on whether it will provide useful information that will sharpen the focus of the product development efforts.

The Contents of the Plan

While the contents of a validation plan will vary from product to product, most plans will include these elements:

- **Validation objectives and test items.**
- **Test user selection requirements.** This is a description of the numbers and characteristics of people to serve as representatives of the user audience. Age, sex, occupation, and type and extent of computer experience are some of the relevant characteristics. For large products, not all test users will be expected to work through all parts of the manual. The

specific assignment of portions of the documentation to test users will be covered in the plan.

- **Site requirements.** This is a description of the room in which the validation will take place. As far as possible, the site should reflect a plausible working environment in terms of lighting, workspace, distractions, and so forth.
- **Hardware and software requirements.** This is a list of the hardware required to run the validation (CPU, memory, mouse, tablet, other peripherals), as well as any additional software (the operating system, for example).
- **Validator instructions and reminders.** This is a step-by-step breakdown of the validation procedure including, as necessary, scripted positioning statements which welcome the test user and inform him or her of the purpose and context of the validation, as well as the steps for leading the user through the various validation tasks.
- **Recording forms.** These are instruments for recording the test user's performance and comments.

Complete the Plan Before Writing Begins

The validation plan is a key element of the manual design and, therefore, should be completed before the writing of the manual begins. In a sense, the plan represents what is expected of the writers, so it's important to include the writers in the development of the plan.

Checklist for Creating a Validation Plan

Here is a list of items that are important in creating a validation plan. You can probably come up with additional items that are important in your company environment.

- Interview technical and marketing personnel to collect information related to product objectives and positioning.
- Analyze internal product and market specifications.
- Work with and learn the product.
- Write validation plan.
- Meet with key project participants to discuss implications of validation requirements on the manual design and writing.

Conducting the Validation Test

The validation test carries out the plan. Here are several points to consider:

Securing the Users

In selecting test users, be guided by the specifications detailed in the validation plan.

Figure 6.
Benefits of Testing for the Writer

Testing frees me to write lean manuals.
I can say the bare minimum about a product or a feature, knowing that testing will reveal areas that need more detail or an in-depth discussion. When I have to revise a 300-page manual and incorporate added features without increasing page count, leanness is a major concern.

Testing verifies the accuracy of my manual.
If representatives of the target audience use my manual and succeed in performing a task, the procedure in the manual works. I can relax and focus on other areas that may need work.

Testing catches errors in my manual.
If representatives of the target audience use my manual and cannot perform a task, I know what needs fixing. Getting this kind of feedback early, when I can do something about it — not four months after the manual has been printed — has obvious advantages for me and for the customer.

Testing helps me learn from my mistakes (and successes).
I can observe patterns of usage, learn what works and what doesn't, and polish my skills based on real information about how customers use my manuals and what they like and don't like. I can generalize from what I learn.

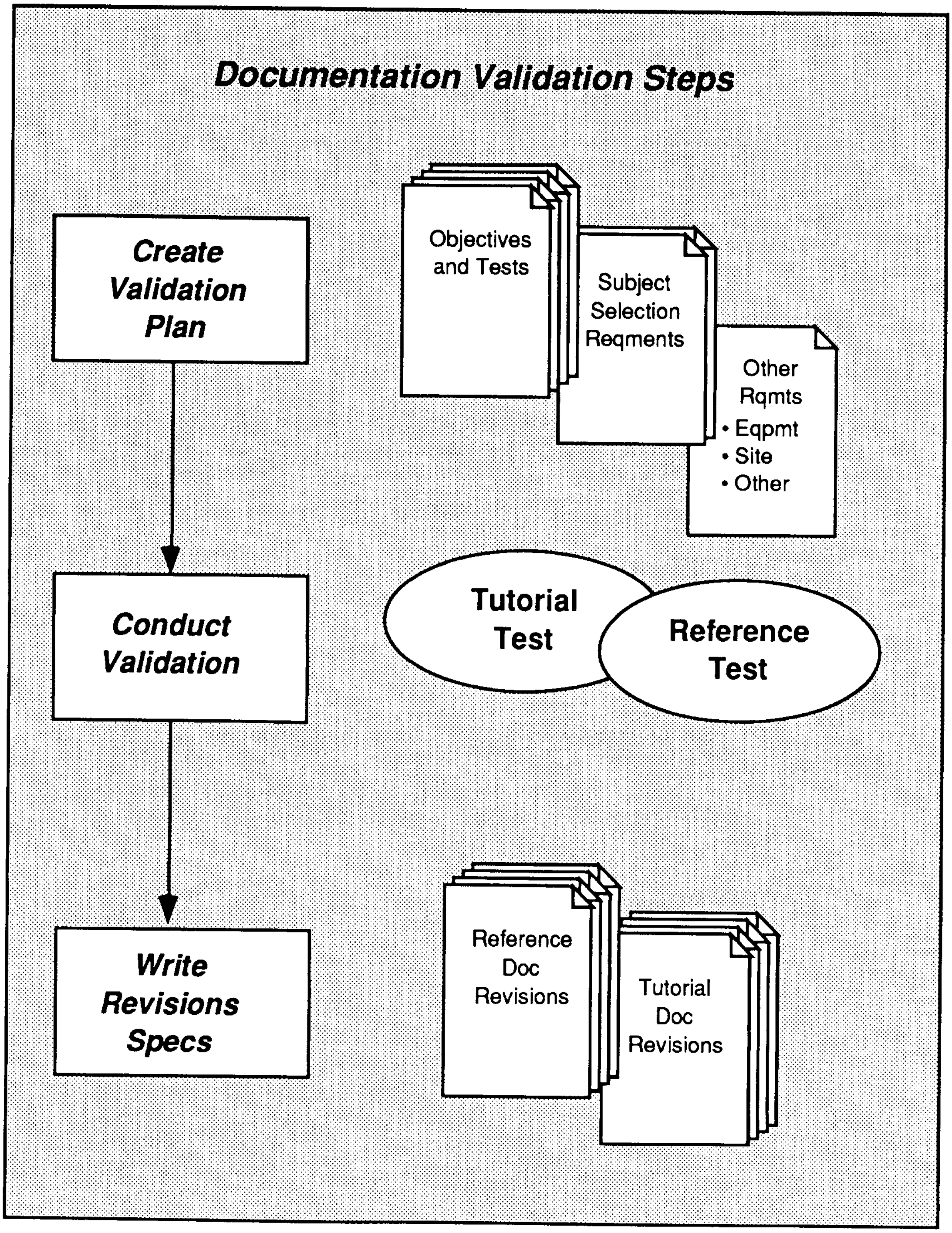
Testing lets me avoid mind-reading.
I don't have to guess what readers want or anticiapte their needs based on an uncanny sixth sense. (Since I can't read minds, not having to try seems like a good idea.)

Testing assures that our customers get better manuals.

Testing gives me confidence and makes me happier, more successful, and more productive in my work.

—*Scottie Zimmerman, Apple Computer, Inc.*

Figure 7.
The Three Steps in Validation



While you might be tempted to find test users within the company, you can get a fairer test from outside people. Your current employees, even those not directly related to the product development effort, can be tainted by the expectations in the local environment.

The Role of Writers and Editors

It is essential that team writers and editors play an active role in the validation test, since nobody knows the manual like the writers. At minimum, writers should observe the testing and, where necessary, probe the test users for insights into confusion and opportunities for improvements.

Ideally, the validator should be the writer. It will be far easier to train a writer to carry out the validator job than to familiarize a dedicated validator with all the ins and outs of a complex set of manuals.

The Validator's Job

The validator welcomes and instructs the test user, and observes him or her working through the required tutorial and reference tasks.

As the test user indicates trouble spots (either verbally or nonverbally), the validator intervenes and gets information about the problem. A draft manual is annotated, and recording forms are completed as required.

A good validator sticks with the test user at every step. He or she is sensitive to all of the grunts, groans, and furrowed brows that signal frustration and confusion. The validator talks to the user, probing for sources of difficulty—how a problem occurred and how it might have been avoided—suggesting improvements, and recording user reactions. Where possible, the validator can make changes to the manual between tests in order to evaluate alternate approaches. Bugs and software interface problems, while not the specific focus of the validation test, are recorded for later transmittal to software test personnel.

Checklist for Conducting the Validation Test

During the actual conduct of the test, keep these points in mind:

- Provide a validation site according to the specifications in the validation plan.
- Provide relevant hardware and software according to the specifications in the validation plan.
- Recruit test users according to the specifications in the validation plan.
- Conduct the validation test according to the validation plan, including welcoming and instructing test users, analyzing user performance, interviewing users, and recording observations and findings.

Writing Revision Specifications

The results of the validation are incorporated into a set of recommendations for revising the manual. Recommendations can focus on organization, content, editorial, or graphic matters. Annotated versions of the manual are also provided with the report.

Checklist for Reporting Validation Results

Here is a checklist that includes some key items in reporting validation results:

- Analyze findings and summarize them in terms of recommended revisions to the documentation.
- Submit a validation revision report and meet with all project participants to review findings.
- Submit reports of software defects (bugs as well as interface defects) to software engineering for disposition.

Summary

In this article, we explored various issues related to empirical validation of computer documentation. We also provided a step-by-step procedure for doing such an empirical validation. With an empirical validation, the real-life analogs of Joe Fisher in User Publications will be able to meet the needs of the real-life analogs of Harry Briggs in Customer Support. And, more importantly, writers will be able to meet the needs of the ultimate end-users. P&I

Neal Margolis directs Skillware, a Silicon Valley firm that relates people to computers and computer products. Neal consults in three areas: training program design and development, technical documentation and user manuals, and software user interface design. In the past 20 years, he has developed materials to teach salespeople, managers, technicians, students, soldiers, and children. **Address:** *Skillware Development, 415 Cambridge Avenue, Palo Alto, CA 94306.*