

Documentation Workflow

Cynthia Shehata

Users hardly ever read manuals; instead they start using the software right away. They are motivated to get their immediate task done. They don't care about the system as such and don't want to spend time getting established, setting up, or going through learning packages. Users often figure out how to apply what they already know to accomplish new goals. There is little desire to explore new functions or to search out information. When a domain expert tries to use a tool designed specifically to support work activities, the orientation is to do real work, not to read instructions or practice step-by-step exercises.

Organize reference material according to real-world goals, rather than system function. Request goal-oriented comments from users during their interactions with the system and base recommendations for structuring reference materials on an analysis of these goals. Take into account actual usage patterns in organizing the material in the manual.

The documentation should be task oriented. Identifying the most common and important tasks, plus the tasks, objects, etc. people had the most trouble with will help make the Help more relevant. To encourage exploration, it is necessary to explain how to recover from errors.

Minimalist instruction reduces the amount of information a user needs to know in order to learn how to use a product.

Questions and concerns workers have when performing a job include

- Why do this?
- What is it?
- What's related to it?
- How do I do it?
- How or why did this happen?
- Show me an example...
- Teach me...
- Assist me...
- Advise me...
- Let me try...
- Watch me...
- Evaluate me...
- Understand me
- How does it work?
- Why does it work like that?
- Compare this or these for me
- Where am I?
- What next?

Table 1: Summary of User Questions

Goal exploration	What can I do with this program?	
Definition and description	What is this and what is it for?	Provide short looped video clips (micons) or provide a list of sample tasks as part of the indexed help message.
Task achievement	How can I do this?	Supply video to be distributed with program, a set of animated cartoons (ex. Part of HyperCard stack), or a set of indexed instructions (i.e. as printed book).
Diagnostic	How did that happen?	Include simple expert system.
State identification	Where am I?	Provide a state transition diagram as a map to show where to go and how to get there. Provide a dynamic map of the last few steps using little screen shots and a record of the command that made each transition.

Minimal Manual In an attempt to attack and support the end-product bias, this manual had an “On Your Own” section. It encouraged users to apply the procedures they had just worked through to new problems of their own choosing. The instructions were left incomplete in order to promote intrinsic interest in the learning process via a challenge.

This tactic of the design competed with the support aspect of the end-product bias by streamlining prerequisites and focusing training activity on the output of real work. In this case, two apparently conflicting strategies were combined to yield a richer design solution.

The design approach for dealing with problems related to the Assimilation Paradox was similar. In its design, a lot of conceptual material was removed. The focus was on concrete procedures.

This was an attack on assimilation through an emphasis on procedural rather than conceptual knowledge. However, this approach was combined throughout with instances of designing for assimilation through the careful use of metaphoric references. The emphasis on procedural rather

than conceptual material reduced the problems learners encountered in achieving the important subskill of getting to a particular area (ex. typing area). The subskill testing also uncovered points at which the basic procedural approach was not optimal.

This material, while it represented a departure from simple procedural descriptions, filled an important need. Without the early qualitative testing, the need may not have been discovered.

The Minimal Manual proved to be substantially faster than the other manuals for the basic topic areas it covered. It also produced equal learning achievement even though it only covered basic topics.

Guided Exploration Cards This approach used cards that were more task-oriented than manuals. Each card addressed a particular functional goal that users could understand on the basis of their understanding of office tasks.

Each card attempted to address its functional goal without reference to material covered on other cards. Each card included specific checkpoint information (to help learners detect and diagnose errors) and error recovery information. There was also a general “What if something goes wrong?” card that described remedies for everything.

Learners using the cards spent less time yet still performed better on a transfer of learning after the test than learners using a commercially developed self-study manual. Taking learning efficiency to be achievement per unit of time, the cards were nearly three times as efficient as the manual. Qualitative analysis of learning protocols shows that the cards increased attention to the task, encouraged learners to explore (successfully), helped learners recognize and recover from errors, and provided a better understanding of learning goals.

User Assistance

Many technologies can be effectively used to support the users of software products. These technologies allow for the proper platform to provide conceptual and reference information, context sensitive stepped procedures, interactive tutorials, frequent on-line updates and much more.

When extensive documentation is required, book-like formats work better than hypertext networks that have less apparent overall structure.

Effective user assistance combines Help, Site Map, Index, and Search functions. The use of these user assistance devices (UADs) is required to support different needs of the user.

While a context-sensitive application programming interface (API) may ultimately be necessary for delivering Help topics on the Web, some straightforward design decisions would improve sites immediately. Ex.:

- use a unique Help page for every instance requiring Help.
- when supporting multiple instances requiring Help with a single Help page, employ mid-page targets.

Web developers could benefit from technical writers writing and organizing the Help content.

On-Line Help

- help messages generated by selecting the desired object. Ex.:
Shift + F1 in Windows
Balloon Help
- context-sensitive help built into application system states or dialog boxes
- generic help text, usually limited in length, available through a help command, menu bar item, function key or icon
- extended help screens, accessible through a “More” button. These can be linked together to form hypertext
- extensive written documentation available on-line, typically to be read like a book.

Documentation

Give users a paper manual. Provide quick reference cards, reference manuals, and tutorials, so people can read the literature that they find most useful.

In the design,

- use high contrast for increased readability
- don't use color alone to convey information
- don't use graphics alone to convey information
- make text readable - 12 point on paper, customizable on-line
- make on-line documentation complete
- make sure that documents convert to SGML cleanly
- bind documentation to lie flat

Style

- Do not use anthropomorphisms -- do not personify the computer.
- Words used in user guidance should be common, meaningful, unambiguous, and complete (i.e., no contractions or abbreviations).
- Avoid computer terminology, unfamiliar, or esoteric terms.
- Use positively rather than negatively worded statements. Negative statements, however, are appropriate for conveying exceptions to rules.
- Use the active not the passive voice.
- Use consistent grammatical construction.
- State messages in short, simple sentences.
- A fact that must be remembered should be at the beginning of the message.
- Place a period at the end of each sentence.
- Use drawings or pictures to illustrate text whenever possible.
- Don't use exclamation points on your command buttons! They make it seem as if the application is shouting at the user!
- In labels or names for menu items, etc., use book title capitalization style. This style is referred to as caps/ lowercase. Capitalize every word except articles (a, an, the), coordinating conjunctions (for example, and, or), and prepositions of three or fewer letters (except when a preposition is part of a verb phrase).

Information should be available

- in product documentation (chapter or appendix)
- in separate documents (on web or available through customer service)

Document accessibility features, tips and warnings for the product.

- accessibility features for users with disabilities
- mainstream features
- tips about how to work around limitations
- keyboard procedures to accomplish tasks
- warnings when the product isn't usable by individuals with certain types of disabilities, or using certain types of accessibility aids

Documentation Checklist

- Easy to search (table of contents, index)
- Focused on the user's task
- List concrete steps to be carried out
- Not too large
- Good writing style
- Bulleted lists
- Highlighted keywords
- Meaningful sub-headings
- One idea per paragraph
- Inverted pyramid style - start with the conclusion
- Half the word count than conventional writing (assign to tech writers)
- Simple explanations of the problems being solved
 - Don't explain every feature in depth. Focus on why the user would want to use the features.
 - Do explain a problem and offer a solution - people will remember it. Don't offer a solution out of context.
 - Step-by-step instructions
- Concepts, not just the features.
 - Well-designed software should be centered on a few, large concepts. Present the concept to the user. Manuals need to cover the task domain, not just the operation of the program
- Effective
 - Try to convey thoughts effectively.
 - Remember that the main function is to teach.
 - Intuitive order and design
- Graphics and screen captures

Manual Testing

A. Scenarios

Get quick & frequent feedback from users

1. Proto-typing
2. Implementation

B. Simplified Thinking Aloud

1. Ask real users to think out loud while performing some typical tasks
Maximum benefit reached between 3-5 users per test
2. Take notes

C. Heuristic Evaluation

User & Task Analysis

Several different people (3-5)

- a. Evaluate usability
 1. average correct
 2. task time
 3. subjective satisfaction
- b. Identify problems
- c. Novice vs. Expert User
 1. Learnability (novice)
 2. Speed of Use (expert) - responsiveness of system, shortcuts
- d. Match between system and the real world
 1. Speak the users' language, with words, phrases and concepts familiar to the user.
 2. Follow real-world conventions, making information appear in a natural and logical order
 3. Help users recognize, diagnose, and recover from errors

D. Evaluate & Refine

1. Evaluate the manual for usability
Get feedback from users who are like the ones who will be using the product
2. Iteratively refine the manual
3. Evaluate the complete early version of the document for usability
 - a. Time to complete a task
 - b. Percentage of task completed
 - c. Percentage of task completed per unit time (speed metric)
 - d. Ratio of success to failures
 - e. Time spent on errors
 - f. Percentage number of errors
2. Iteratively refine the documentation

II. Release & Follow Up

- A. Plan and implement the introduction of the product to users, including final usability evaluations to ensure that the product has met the goals established at the beginning of the process
- B. Create and monitor feedback mechanisms to gather data for future releases

References

- *Ameritech Graphical User Interface. Standards and Design Guidelines: Output to User.* <http://www.ameritech.com:1080/corporate/testtown/library/standard/guix5.html#5.3.1>.
- Bad Designs. Common Questions and Answers. <http://www.baddesigns.com/faq.html#next>
- Carliner, Saul. "Part I: What Electronic Performance Support Systems Are (and Why They Are Important to Technical Communicators)." <http://www.tds.co.il/epss1.htm>
- Carroll, John M. and Mary Beth Rosson. *Interfacing Thought: Cognitive Aspects of Human-Computer Interaction.* Cambridge, MA: MIT Press, 1987. Chapter 5: *Paradox of the Active User.* <http://www.cs.vt.edu/~rosson/papers/paradox.pdf>
- Human Computer Interaction. http://www.dcs.napier.ac.uk/~michael/hci_ohps/hci41n.html
- Interface Hall of Shame. "Globalization." <http://www.iarchitect.com/global.htm#MOREINFO>
- Interface Hall of Shame. <http://www.iarchitect.com/clarity.htm>
- *Macintosh Human Interface Guidelines.* Addison-Wesley Publishing Company, 1995.
- Microsoft Accessibility. "User Documentation." <http://www.microsoft.com/enable/dev/ue.htm>
- Nielsen, Jakob. "Guerrilla HCI: Using Discount Usability Engineering to Penetrate the Intimidation Barrier." http://www.useit.com/papers/guerrilla_hci.html
- Nielsen, Jakob. "How Users Read on the Web." <http://www.useit.com/alertbox/9710a.html>
- Nielsen, Jakob. "The Paradox of the Active User." <http://www.useit.com/alertbox/activeuser-paradox.html>
- LUCID Framework Overview. <http://www.cognetics.com/lucid/lucid.html>
- Preece, Jenny et al. *Human-Computer Interaction.* Addison-Wesley 1994. 308-318.
- "Principles of good GUI Design." http://axp16.iie.org.mx/Monitor/v01n03/ar_ihc2.htm
- Requirements Capture. http://www.dcs.napier.ac.uk/~michael/hci_ohps/hci444r.html
- Tognazzini, Bruce. "How to Publish a Great User Manual." <http://www.asktog.com/columns/017ManualWriting.html>
- Welinske, Joe. "Piecing Together the Online Help Puzzle A Pundit's Perspective." <http://www.tds.co.il/jwpuzzle.htm>
- Welinske, Joe. "A Study of How User Assistance is Employed in Web Sites." <http://www.tds.co.il/jwwebhelp.htm>