Essential Elements of Technical Communication

Chip Jones

1/20/15

What’s “Hot” Today in TC/UA?

- Content management
- Mobile & tablet, app support
- Videos replacing text
- Social networking to improve user analysis
- DITA and other structured writing paradigms
- UA with UX (see http://conveyux.com/agenda/)
- Agile development

What’s Losing Out?

- User & task analysis
- <u>Real</u> usability testing
- Writing/editing quality

What Gets You Hired?

- Strong writing and grammar skills
- Grasp of the writing process and the best practices of technical communication
- Focus on user needs
- Inquisitiveness to learn the product you’re documenting with minimal assistance
- Attention to detail (proof/test everything)
- Ability to learn software tools quickly
- Ability to work both collaboratively and independently

Writing Process

1. Understand product/feature & why it’s used
2. Understand the audience & how they’d use it
3. Perform detailed task analysis
4. Plan project, communicate plan
5. Write content iteratively
6. Test deliverable by itself and with the product
7. Evaluate the released deliverable and plan for improvements

User and Task Analysis

- First step in implementing a minimalist approach
- Crucial for determining what information the user needs and how much detail is required

User Analysis - Observation

- Learning about ordinary users by observing them in action. Why not ask:
  - Focus groups away from work?
  - Expert users or managers of users?
  - Users themselves?

User Analysis – What We Learn

- Goals – what they are trying to achieve
- Tasks – what they do to achieve these goals
- Which tasks they perform often
- Personal, social, and cultural characteristics they bring to the tasks
- Environmental influences (where work is done)
- Prior knowledge and experience level
- What success looks like to them
- Where they look for help when things go wrong

User Analysis – How to Do It

- Understand your target market
- Identify representatives of that market
- Determine what information you want to gather
- Gather the information
  - Interviews
  - Surveys (questionnaires)
  - Observation (best)
- Analyze your findings
- Create one or more personas based on the findings

User Analysis – Working with Personas

- Fictitious users based on your user research
- Develop (or borrow) personas with real names, roles, personalities, motivations, and even a photo
- Personas help keep you focused on proper amount of content to meet the persona’s info needs
- The writing team and internal partners share the same personas for consistency
- For more robust products, develop multiple personas to represent different user types

Task Analysis

- What will the users do with the software, why will they do it, when, and how?
- Think “tasks and knowledge topics” (Info Mapping)
  - IM is not minimalism per se – must avoid conceptual bloat
- Avoid basing the task analysis entirely on the screen content (tasks can span multiple screens; not everything on a screen needs to be documented)
- With data collected, build a user/task matrix and validate it
- Validate again in usability testing and from user feedback

But, we…

- …don’t have enough user information and/or time to do good user and task analysis (especially new products).
- …need to start content development before we have holistic picture of new features and impacts (and user experience, user needs, user tasks).
- …are getting specifications on page-by-page basis and have to develop help that way.

So we will…

- …use what information we can get from Product Marketing and other user research (including existing users) to create personas to guide us.
- …develop what we can when we can, working iteratively and revisiting content and structure as more information becomes available.
- …focus on user tasks when drafting content, even if we must—for logistical reasons—begin our development on a page-by-page basis.

“Real” Usability Testing

- Asking a friend to see if the instructions match the product?
- Demonstrating some of your content for clients in a WebEx call?
- Putting sample users in front of your product with your documentation, asking them to complete some tasks, and observing what they do and ask about?

Key Elements of Writing & Editing

- Task Orientation with Minimalism
- Clarity
- Style

Structured Writing

- Dates to Bob Horn in early 1960s
- “A publishing workflow that lets you define and enforce consistent organization of information in documents, whether printed or online.”
- A file—either a document type definition (DTD) or a schema—captures and enforces the defined content rules (headings, lists, etc.)

Structure: Rockley’s 5 Principles

- Chunking (or use of information blocks)
  - Not a paragraph: no topic sentence, no fluff
- Labeling – every block has a meaningful label
- Relevance – just the facts, ma’am
- Reusability – same content, multiple uses
- Consistency – “For similar subject matters, use similar words, labels, formats, organizations, and sequences.”

Minimalism

- Dates to research by team led by John Carroll at IBM in 1980s
- Famous book: *The Nurnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill (1990)*
- Book originally written for course designers
- Principles now supported by the major authoring tools as well as XML editors

#1 – Use an Action-oriented Approach

- Provide an immediate opportunity to act, but goal is “guided discovery”
- Avoid any extraneous information [just enough]
- Present most conceptual info after the fact—when users are more motivated to use it (i.e., after user has had a chance to explore and even make errors)
- Don’t interrupt the user’s flow by forcing him or her to access help information

For example…

- Avoid lengthy introductions.
- In a task topic, provide a very brief context if it is needed, but don’t provide a detailed explanation of what the task is, etc. If you think novice users will need that level of information, write a concept topic and link to it from the task topic.
- In general, document only one way to do something. If it seems absolutely necessary to alert users to alternatives, just mention them and let the user explore/discover how to use them. Don’t document alternatives in detail.
- If in doubt about “extra” information in a task topic, link to rather than include it—so that the user can keep “doing” rather than reading (unless he/she chooses otherwise).

#2 – Focus on Real Tasks

- Don’t teach how the <u>system</u> works; teach how to solve real domain goals using the system
- Less is more: provide only hints and tips to guide exploration, not exhaustive detail
- Eliminate sequence—design topics to be read in any sequence

For example…

- In general, write task topics first. Add conceptual, reference, or other non-task-based content and link to it only as necessary to support successful completion of tasks.
- Avoid documenting features, pages, screens beyond what is necessary to support a given task.
- Remember the importance of goals when analyzing what tasks user needs/wants information about.
  - Goal of user is rarely to learn everything there is to know about a page or multiple ways of doing a task.
  - Goals are more likely about finishing a job or task quickly, successfully, and/or with a minimum of frustration. Other goals might focus on satisfying customers or supervisors, getting out of work on time, etc. Most goals are accomplished by completing tasks.

#3 – Support Error Recovery

- Prevent mistakes with good design and thorough testing
- Expect users to make mistakes, and then support learning from them
- Ensure the error information supports detection, diagnosis, and recovery
- Provide error information on the spot with instructions for resolution if necessary

For example…

- Use all of your assets/content types to do this, including:
  - System messages
  - Embedded text
  - Field help
  - “Page-level” help (that is, help displayed in main help window)

#4 – Support Reading to Do, Study, and Find

- Acknowledge disparate user goals
- Be brief—don’t spell out everything
- Provide closure for chapters or groups of tasks

For example…

- Keep content in each topic to a single rhetorical purpose. For example, avoid having conceptual information in your task topic.
- Write in a modular way, so that each topic is self-sufficient and makes sense regardless of user’s reading sequence, prior knowledge, etc.
- Cover one appropriately sized, coherent, closely-related subject per topic. Avoid overly lengthy topics. Eliminate extraneous information.

Risks of Minimalism

- From David A. Farkas, “Layering as a ‘Safety Net’ for Minimalist Documentation”
- User may be unable to successfully complete the task
- User may complete the task but expend more time and energy than intended to do it
- User may develop an incorrect mental model of system, leading to later difficulties

DITA: Key Concepts

- An XML-based architecture for technical writing
- Created by IBM, made public through OASIS
- Facilitates topic-based writing versus book-based writing
- Enables separation of content from presentation
- Enables reuse in a wide variety of information deliverables
- Allows specialization and inheritance
- A “database of topics”

DITA: Benefits

- Works well in a web application development environment
- Source topics can be stored and updated in a CCMS that’s separate from the application code
- Source topics can be updated “on the fly”
- Help topics can be rendered dynamically
- User assistance presentation can be controlled by development based on UI standards
- Tech writers can author a wide variety of other deliverables from the same source topics

Clarity
Overview

- Goal: readers understand words the first time
- Good writing requires revision & iteration
- Critical when your words will be translated or localized

Clarity
Focus on the Meaning

- Avoid or replace:
  - Imprecise words
  - Intensifiers (very, just, actually, fairly, etc.)
  - Long sentences
  - Long paragraphs
  - Wordy phrases (in order to > to)
- It always takes longer to write well

Clarity
Other Key Points

- Write consistently
- Avoid noun strings
- Write positively – tell what to do, not what not to do, if possible
- Use technical terms only when necessary
- Define new terms on first use

Layering

- Providing different levels of content for different use cases
- Revealing more details gradually
- Supports task vs. system perspective
- Provides support for different stages of use
- Options
  - Progressive disclosure
  - Extended examples
- Best for concepts, definitions, and basic/advanced definitions

Style

- More than just formatting, it’s a critical way of making your work professional
- When you’re careless with grammar, spelling, etc., your readers trust you less

Sentence Fragments

- Fragments are simply incomplete sentences
- Can be appropriate or not
- Examples:
  - Find the page. In the book.
  - At the end of the street.
  - He was tired; the end of a long day.

That vs. Which

- Area of much confusion among English speakers and writers
- It’s a matter of essential vs. nonessential clauses = can you safely omit the clause without changing the meaning?
- *That* always introduces an essential clause, but *which* can be used with both types.
- Essential: **The painting that was hanging in the foyer was stolen.**
- Nonessential: **The painting, which was hanging in the foyer, was stolen.**

Active vs. Passive Voice

- Active voice clearly identifies the actor or agent of a sentence. It is almost always preferred in technical writing.
  Ex: Users enter codes to identify dates.
- Passive voice uses forms of the verb "to be" (is, was, has been, being, etc.) and often doesn't identify the agent. Look for the words "by" or "for" in passive sentences.
  Ex: Codes <u>are</u> entered *by* users.

Tense

- Technical communication works best when you use the present tense unless the action clearly occurred in the past or will occur in the future.
- Examples:
  Present: Word displays the date.
  Past: Word displayed the date.
  Future: Word will display the date.

Mood

- Indicative mood states facts and opinions.
  Ex: Microsoft Excel helps you create spreadsheets.
- Imperative mood instructs or commands. It tells someone to do something.
  Ex: Enter the date.
- Subjunctive mood expresses wishes, doubts, or desires and is rarely appropriate in technical writing.
  Ex. If I were a rich man, I'd retire now.

Style Guideines

- Three levels:
  - Discipline or industry guides, such as *Chicago Manual of Style,* Microsoft (see http://faculty.washington.edu/farkas/TC407/MSTP-V3.pdf) , Sun, etc.
  - Corporate standards
  - Project-level standards, often called style sheets (not to be confused with cascading style sheets or CSS)
- If you don't work for a company that has one, consider developing your own, <u>even as a lone writer</u>