

## **ONLINE SUPPORT SYSTEMS: TUTORIALS, DOCUMENTATION, AND HELP**

Stuart A. Selber, Clarkson University

Johndan Johnson-Eilola, Purdue University

Brad Mehlenbacher, North Carolina State University

### **CHAPTER 94**

**CRC HANDBOOK OF COMPUTER SCIENCE AND ENGINEERING**  
CRC PRESS, BOCA RATON, FLORIDA, 1997, 1619-1643.

### **INTRODUCTION**

**Online support systems** help computer users achieve goals and accomplish tasks within the contexts of their **primary work**. Although this definition is extremely broad and includes a wide range of digital forms—from low-end interface elements to high-end hypermedia applications, in this chapter we generally focus on planning, designing, and testing mid-range systems: **tutorials, documentation, and help**, regardless of their virtual instantiation. We discuss electronic rather than print-based forms because organizations increasingly deliver user support online for a variety of reasons: to reduce development and production costs; to anticipate distributed computing systems and other environments in which users rarely have easy access to print-based materials; and to benefit from the sophisticated searching and interactive capabilities that online environments can provide. In cases where print-based support is still necessary (for example, in packing instructions and in some troubleshooting areas), processes for constructing these documents can be extrapolated from the discussion that follows.

Determining what types of online support to provide users is a deceptively complicated task: they often work with computer technologies in a variety of contexts and with numerous purposes and goals, and single users frequently bridge more than one context, purpose, and goal over time. Consider a technical communication group that has purchased a desktop publishing program to help

them write, design, and publish documentation. Not surprisingly, they might find useful support in a wide variety of forms. An introductory, online tutorial might guide them through basic program functions such as creating files, making templates, and importing graphics. This same tutorial, which might also include an accompanying workbook, could encourage them to examine case examples to learn how discrete portions of the program work together. While using the desktop publishing program itself, the group might employ context-sensitive help to see brief descriptions of available tools. Under certain conditions, they might rely on more extended online documentation for feature or process overviews. Still, in other cases, they might link from the online documentation to brief tutorials that model particularly complicated procedures.

This scenario illustrates the connections among and differences between the three types of online support we discuss in this chapter. Tutorials include the broadest possible topics, with users learning about features and tasks by engaging some combination of explanation, example, and hands-on experimentation. Online documentation has a narrower pedagogical scope, with users normally consulting reference information for overviews or assistance with task-oriented procedures. Online help usually has the narrowest focus, with users needing to solve particularly pressing problems as quickly as possible and with minimal interruption.

#### **UNDERLYING PRINCIPLES**

Developing useful online support is a complex, recursive process that should parallel product development in central ways and situate planning, designing, and evaluating as non-linear, rhetorically-based activities. Before beginning this recursive process, at least three foundational areas merit serious attention: the advantages and disadvantages of online support; the differences between online and print-based support; and the rhetorical frameworks useful for developing online support—users, goals, and time/space constraints.

## **Advantages and Disadvantages of Online Support**

There are both advantages and disadvantages to creating and delivering support online. The advantages for designers are increasingly clear. Compared to print-based genres, online support is relatively inexpensive to produce, distribute, and update once initial procedures have been stabilized and adopted by project teams. And, because all forms of online support require shorter production cycles than their print-based counterparts, designers can continue working on content issues until just before final products are released (usually one to two weeks). This expanded development period is particularly useful for accurately documenting task-oriented information. The efficiency gains afforded by mature development cycles for online support are ultimately noticeable in five key areas: worker productivity, task completion time, the overall time needed for system development, the materials and economic resources needed for production, and maintenance activities such as creating and distributing revisions [Petrauskas, 1991].

Users will find that well designed online support can assist their learning goals and task objectives in substantial ways. Obviously, large amounts of reference or database information are more easily and personally used in conjunction with robust software engines that automate or augment search routines. But, beyond the keyword and pattern searching of vast textual and numerical landscapes, the potentially dynamic and interactive nature of online support offers many distinct advantages. For example, in online tutorials—which are often considered a form of Computer-Assisted Instruction (CAI), Computer-Assisted Learning (CAL), or Computer-Based Training (CBT)—users can engage highly interactive, self-paced educational activities at their own convenience and pace, either for immediate performance improvements or for general professional development. In online documentation environments, users can usually choose from among many different organizational patterns and navigational systems, as well as customize

instructional content by adding personal annotations, trails (via bookmarks), and entirely new texts. And, in online help systems, users can stay focused on their primary tasks by employing tightly integrated, context-sensitive assistance at particularly pressing impasses. In addition, all three forms of online support can exploit the artificially intelligent operations of computers: in at least basic ways, certain types of online support can track user actions and productively respond by adapting what's available or prominent at any one moment.

The potential disadvantages of online support, however, are not insignificant. As we discuss below, many of these disadvantages relate to the physical and rhetorical differences between pages and screens and between common ways of instantiating print-based and online materials. In addition, there are other potentially problematic issues to consider. Novice users struggling to learn new applications may find little comfort in assistance that is similarly provided online. Users working with the minimum hardware and/or software requirements needed to run applications may experience performance problems that discourage them from accessing support or from taking advantage of its advanced features. And, poorly designed systems, while conveniently available online, are often inferior in form and function to well designed, print-based genres of assistance. Although these disadvantages should not discourage developers from creating online support, they should be considered during planning and designing stages.

### **Differences between Print-Based and Online Support**

Important physical and rhetorical differences differentiate print-based and online support. Too often, these differences are simply ignored or overlooked, a mistake often resulting in two unfortunate situations: support that fails to exploit a system's unique abilities to store, structure, and retrieve digital information; and/or support that applies those unique abilities in ways that are inappropriate to the needs of users. Designers converting print-based support to online support may be tempted

to simply dump their source files online without significantly rethinking core design decisions. Research indicates, however, that such time-saving approaches can be disastrous for end users [Rubens and Krull, 1985]. In general, the physical differences between print-based and online support relate to resolution, display area, aspect ratio, and presence, while the rhetorical differences relate to organizational, navigational, and contextual structures. These differences are summarized in Tables 1 and 2.

The *physical* dimensions of online support provide a writing and reading space that is qualitatively different than that of pages. Computer screens are much harder to read, for example, because their resolution is typically much lower than professionally printed materials and because the area available for displaying information is often much more limited. Available space for displaying multiple documents simultaneously is also much more limited for screens than printed pages (screen territory versus desk space). These two constraints, moreover, can limit certain aesthetic dimensions of document and graphic design. In addition, the 4-by-3 aspect ratio of many computer screens—the ratio of their horizontal to vertical dimensions—more closely resembles a movie screen or television monitor than printed page. This expanded horizon can encourage developers to include more information than they probably should for good readability, and it clearly complicates design principles originally developed for print-based, portrait orientations. The presence of text on screens also differs from that of pages in dramatic ways. While the information on pages is static and immutable, the information on screens can be dynamic and interactive, as well as mutable. Importantly, this malleability provides unique opportunities for providing customization opportunities and for practicing **user-centered design**.

The *rhetorical* dimensions of online support—organizational, navigational, and contextual structures—influence the ways in which users understand system logic. Although online support is often developed using organizational structures familiar to print-trained readers (linear and hierarchical, for example), online support systems can employ dynamic and associative (web-like) structures that allow users to organize and reorganize information in a wide variety of ways and move between many different levels of instruction. Moreover, online support systems can adapt dynamically to user performances, shifting goals, and changing contexts in ways that the technology of print makes impossible. Readers may find that certain navigational structures such as bi-directional links, history trails, and overview maps are similarly more dynamic and at least initially unfamiliar, although many navigational structures such as bookmarks and indexes have been appropriated from the realm of print. The idea of both physically and metaphorically navigating through online information space, however, departs radically from the user passivity often associated with writing and reading more traditional, printed texts.

Because online support is often more dynamic and unfamiliar, its contextual structures can be poor. While using online support, individuals lose many traditional user cues—page numbering and tabs, for example—that allow them to work efficiently and effectively with texts, and they can also lose the spatial comfort that holding a printed book can provide. In fact, as [Haas, 1989] reports, users often have trouble “getting a sense of text,” that is, seeing meaning and structure, in extended online information spaces. And, she notes that writers commonly struggle with formatting, proofreading, and reorganizing texts that are solely created and examined online.

## Rhetorical Frameworks for Online Support

In developing online support, designers must carefully consider the needs and wants of users, including their previous experiences and abilities, their short- and long-term goals, and the environments in which they work. Such a framework for online support carefully considers two important areas: the *rhetorical* issues of users, goals, and time/space frames; and the *formal* characteristics of tutorials, documentation, and help.

The main objective of online support systems, regardless of their form, is to help users achieve *goals* as they negotiate the very real constraints of various *time/space frames*. In fact, these terms—users, goals, and time/space frames—represent key elements in understanding and designing useful systems (see Table 3). In working with Table 3, remember that many different forms of online support may be necessary to adequately help users work across multiple contexts. In addition, each rhetorical element commonly affects the other two: for example, if a novice user's goal is only to complete a task (and not to understand its broad implications or varied uses), then online help may be the best solution even though the user will not gain more conceptual understanding. Warning symbols and emergency instructions are two such cases.

Our approach to online support encourages designers to consider the rhetorical aspects of Table 3's left column—users, goals, and time/space frames—rather than the formal, surface-level characteristics of its top row—help, documentation, and tutorials. Approaches that begin with formal document characteristics rather than user contexts are often too brittle to succeed in real-world situations. User actions rely heavily on contingent, situated, recursive actions rather than acontextual plans [Beabes and Flanders, 1995; Boy, 1992; Suchman, 1987; Winograd and Flores, 1987]. When the development of online support systems begins with a focus on the

formal characteristics of different forms of communication rather than the contexts and purposes of use, the resulting systems can appear well designed and executed but generally fail to address the needs and constraints of users; sometimes, the resulting systems force users to remake their own work processes and even long-term goals in order to adapt to the forms of the online instructions [Johnson-Eilola, in press]. James Paradis [1991], for example, analyzes operator's manuals for construction tools to illustrate how decontextualized instructions purchase simplicity and clarity at the cost of human lives. Although there are numerous excellent resources for developing specific forms of support [Carroll, 1990], designers should understand that such works often assume previous rhetorical analyses have lead developers to that specific form. By carefully analyzing the rhetorical aspects of users, goals, and time/space frames, designers can work to avoid such situations and provide online support that is both more appropriate and useful.

### Analyzing Users

The three types of online support—tutorials, documentation, and help—each addresses different kinds of users, including those individuals whose skill sets change from one activity to another. The traditional terms “expert,” “intermediate,” and “novice” can be misleading because there are many different ways to measure someone's knowledge: for example, by specific application, platform, task, and profession. Physicians using a new diagnostic support program may be novices in terms of this specific application, but experts in terms of medical concepts, terminology, and even similar applications. They might need brief, on-screen definitions to remind them of potential drug interactions, but longer-term, intensive tutorials (including simulations and self-administered quizzes) to help them restructure patient interviews in ways that the new software program can productively support.

Experts (of whatever type) typically benefit the most from online help because they already possess rich structures of understanding; when they ask for assistance, it's often for help in retrieving information from long-term memory. For example, error messages in many Unix versions also function as online help by reminding expert users of complicated syntax. Novices, however, usually require more structure and learning aids to understand basic concepts and specific commands and to build mental models on which they can later draw (experts would find such assistance tedious). Questions to ask when considering user types include the following:

- How much do users know about the software?
- Do users know similar software?
- How much do users know about their goals and tasks?
- How much do they need to know about their goals and tasks?
- Do they need information for short-term (consultative) or long-term (memorized) uses?
- Do they need to learn background information or concepts?
- Are there some areas in which users are experts but others in which they are novices?

### Defining and Redefining Goals

Designers should consider the goals of users before developing online support systems. In some cases, users in the middle of a complex task may only need minor information to successfully complete their work. For example, an intermediate or expert user of a multimedia environment might, during a large-scale authoring project, need to select a transition effect when splicing two segments of digital video together. A brief, online list of available effects and their syntax would constitute appropriate support, providing quick assistance without much cognitive overhead.

A new multimedia user, however, might be better served by an interactive tutorial or long-term training session that discusses not only functional commands, but the basics of screen design, graphic arts, music, and learning styles. In the former case, the specific user goal is to code a transition from one video segment to another, a fairly low-level task. In the latter case, the user goal is not merely to get information about transitions, but to learn a broader range of multimedia concepts, of which transitions are only one small part. The information about transition effects referred to by the expert might even be reproduced as part of a lesson for the novice, but with a stronger, pedagogical framework.

When researching the goals of users, these goals must often be divided into subgoals and subtasks appropriate to their current knowledge bases and purposes [Card, Moran, and Newell, 1983]. Often, a support suite must help convince users that their immediate goal cannot be accomplished without first undertaking some broader learning. Furthermore, when the goals of users are fairly ambitious but tutorials are required, users may need motivation to keep their attention level high. Such motivation can range from clearly stated objectives that help users track their progress during lessons to institutional rewards for completing a series of lessons. Questions to ask about user goals include the following:

- What, exactly, do users want to accomplish?
- How might those goals be supported by the software/hardware?
- Are there conceptual issues that users need to learn before using the software/hardware to achieve their goals?
- Should user goals be divided into more manageable sub-goals, tasks, or sub-tasks?
- Will users need feedback (screenshots, audio acknowledgments, and so on) to know that they have achieved their goal?

- Will users need motivation?
- What kinds of motivation might be appropriate?

### Locating Support in Time and Space

Another crucial rhetorical element is the amount of time and screen space available to end users. For example, in word-processing programs, the workspace is normally the screen into which texts are typed and displayed. The workspace is, therefore, the focus of user attention, and the difficulty of re-focusing that attention must be taken into account when determining the type of online support to offer. In fact, focus broadens and changes as users move from online help (which tightly integrates support and exists only briefly outside the workspace), to online documentation (which provides on-screen text paralleling the workspace), to tutorials (which provide a simplified learning workspace, or which construct a learning space separate from the workspace and only infrequently refer to it during exercises or testing). At times, some users may resist redirecting their attention from the workspace to a learning space, even when that action is well worth the effort. As one might expect, this resistance can relate to the knowledge levels and goals of users. For example, individuals who are experts in programming but novices in a specific application may resist attempts to re-focus their attention away from the workspace and toward a tutorial during pressing primary work.

Questions to ask about locating support in time and space include the following:

- How much re-focusing is required for users to engage the support?
- Do they have both the time and energy to re-focus their attention?
- Would users benefit from a simplified version of the workspace for learning?
- Is there sufficient room for on-screen, parallel spaces, or would a

workbook or paper manual be required?

- Can users be convinced to re-think or broaden their goals to accommodate new foci?
- Are there other, environmental factors that will interfere with some types of support (lighting, noise, technological limitations, and so on)?

## **BEST PRACTICES**

Generally, best practices for developing online support can be divided into three broad stages: planning, designing, and evaluating. During planning stages, developers conduct needs assessments and create specifications. This activities provide the necessary foundation for designing online support, a process that must address both global issues (such as organizing and maintaining systems) and local issues (such as chunking information and providing discourse cues). Evaluating stages include both formative testing—in-process evaluations that may lead to redesign, and summative testing—post-production evaluations that measure end quality [Duffy, Palmer, and Mehlenbacher, 1992]. These three stages often occur recursively, as decisions or responses in one area affect the other two.

### ***Planning Online Support Systems***

Planning is a critical task in the recursive process of developing online support systems. Early decisions about form and content will centrally influence subsequent design stages and final products, as will determinations by managers and others about individual and group responsibilities. In fact, although there are many reasons why online support systems fail or are only partially successful, in many instances these reasons can be traced to poor or inadequate planning [see Caernarven-Smith, 1990, for an annotated bibliography on the benefits of early

planning practices]. Indications of poor or inadequate planning during development stages include incomplete style and design guidelines, overlapping work duties, missed milestones and deadlines, and significant deviations from standard production practices.

Unforeseeable events will certainly alter schedules and plans in unpredictable ways. Nevertheless, it's not uncommon for thoughtful designers to spend 25% of their total project time on planning issues: planning opens lines of communication between managers, designers, and users; coordinates the tasks and activities of project team members; encourages systematic approaches to online support; and, in the long run, saves both human and economic resources.

Assuming that management has already conducted feasibility studies and established the market viability of new products, at least two essential tasks relate to planning online support systems: conducting needs assessments and developing specifications. A wide range of artifacts often inform these two stages, and are therefore collected before planning begins—among them, feasibility studies, business plans, product requirements, cost/benefit analyses, mission statements, user surveys, intelligence (research on competitive products), demographic information, and other market research.

### Conducting Needs Assessments

Needs assessments precede specifications and help determine both external (user/customer) and internal (designer/manager) requirements for online support systems. For new products, these assessments address broad-based questions: for example, *What should our online support systems accomplish? What kinds of online support systems can we realistically deliver and maintain? What substantial constraints are imposed by the working environments of users?* For existing products, the scope of these questions narrows considerably: for instance, *What*

*features are the most and least useful? What features require the greatest amount of customer support? What features might be added to or eliminated from the next version?* For both external and internal purposes, needs assessments help clarify the main objectives and roles of online support.

Like all research projects, needs assessments begin with questions—like the broad and narrow ones listed above—that derive from real organizational, business, and social contexts, both external and internal. Use these questions to help clarify the goals of needs assessments, which should be stated in clear and precise terms.

Next, select empirical methodologies—qualitative and/or quantitative—that can guide your research questions in systematic ways. Useful modes of inquiry include interviews, surveys, user observations, and case studies. Notably, field work in the form of interviews and observations are particularly helpful for understanding the perspectives of users because it occurs within their natural (working) environments [Beabes and Flanders, 1995].

Interpret the results of your research carefully and modestly, keeping the results gathered from any single inquiry in perspective. For higher reliability, use a variety of methodologies that allow you to examine problems or issues from many different perspectives: patterns suggested by multiple approaches are generally more reliable. Also, for further confidence in your findings, compare your results with those discussed in any related literature. As with all research projects, fully document and share your results in a completion report. Meeting face-to-face with team members as this report is distributed will allow you to answer questions about your methodologies and findings, as well as limit any possible misreadings of the conclusions.

Task analyses are often an integral part of a thorough needs assessment. They identify the complex range of actions that users must perform, and they further

divide these actions into subactions that the planned online support systems should discuss. From such task analyses, designers gain a clearer sense of one important (and constantly evolving) triangulation: the information needs of users, their work contexts, and their knowledge bases. By considering the relations between these three areas, designers can better serve as user-advocates and representatives of other external interests. Moreover, these activities provide designers with opportunities in which to learn and influence products.

In addition to task analyses, needs assessments also often include some form of media analysis that determines what types of deliverables should be developed in connection with a project: tutorials, documentation, and/or help, paper-based and/or online. Also, because they help articulate form and content, needs assessments can suggest the kind of interdisciplinary team that should be formalized. Although internal projects are often handled by programmers and other technical specialists, external projects will require designers who can write for online information spaces, edit for both developmental and mechanical concerns, manage online projects, test the usability and quality of support, understand user psychology and cognition, design human-computer interfaces, design instructional materials, market products, and so on.

Finally, needs assessments focusing on internal issues can help establish budgets and timelines. Determine what you can afford to design, maintain, and update, and what you can productively accomplish in parallel with product development [see Smith, 1994, for approaches to and worksheets for estimating project costs]. This information is repeated in specifications for online support.

### Developing Specifications

Specifications, also commonly called documentation plans or blueprints, outline courses of action for developing online support systems, including both product and

process descriptions. Product descriptions detail the rhetorical and physical dimensions of deliverables: feature summaries, audience profiles, and system organizations are just some of the areas you should discuss. Process descriptions outline approaches to accomplishing work, including development and production schedules, review protocols, and testing procedures. Although both products and processes may change in a variety of ways, developing specifications is an important planning task: in these documents, you must be explicit about all the predictable aspects of a project.

There are other important reasons for developing specifications. First, they can help you anticipate problems before they occur and outline workable solutions and contingency plans. They can also help you understand the consequences of any dependencies and risks. Second, specifications can help you articulate the complexity of your task and therefore establish realistic expectations about what can be productively accomplished during development cycles. Third, because specifications include contributions from all team members, they provide early opportunities in which managers, subject-matter experts, and others can provide necessary feedback, express concerns, and raise doubts. In turn, such early participation solidifies both individual and group responsibilities, and it generally increases an organization's commitment to the successful completion of deliverables.

Although there are many different ways to develop specifications, the following 12 elements should always be included:

- Background Information
- Audience Profiles
- Topic Outlines
- Detailed Outlines
- Style/Design Specifications
- Project Responsibilities
- Production Specifications
- Review Protocols
- Development Schedules
- Cost Estimates
- Dependencies and Risks
- Sign-Off Sheets

### Background Information

Include overviews of what you are documenting and what you are developing (their main uses and features), goal and mission statements reflecting managerial, developer, and user concerns, immediate and long-term action plans, scope and purpose statements, instructional goals, the results of your needs assessments and task analyses, common scenarios and contexts in which individuals will use your deliverables, discussions of what your deliverables will not include, associated documents and products, possible authoring tools, and definitions of any specialized language.

### Audience Profiles

Move beyond simple demographic information to include user contexts, learning styles, and motivations. Describe the nature of their work, their concerns and biases, their skills both in using computers and in understanding the subject matter you are documenting, what they expect from online support systems, their interest levels, their knowledge bases, and any presuppositions they may have about what your support system ought to include.

### ***Topic Outlines***

Provide high-level content overviews, including module names, section names, and first-level headings. Include storyboards that map information flow and system logic. Estimate the total number of screens the support will include. Whenever possible, build a system that allows extension and modification without elaborate revision at the global level.

### ***Detailed Outlines***

Expand your topic outline to include any front matter, submodules, subsections, scenarios, examples, index terms, and appendices. Fully describe the contents of all

planned information, both textual and graphical. Provide sample sections and example screens.

### ***Style/Design Specifications***

Discuss organizational patterns, navigation aids, reference aids, interface designs and grids, linking strategies, aesthetic dimensions, uses of graphics, user cues, emphasis markers, text conventions, style guides, and other standards. Provide thumbnail sketches and sample screens.

### ***Project Responsibilities***

List all team members, their titles, and their roles and primary and secondary responsibilities. Identify project leaders for each major phase of development.

### ***Production Specifications***

Describe the final physical forms of all deliverables. Consider disks, disk labels, release notes, notes for CD-ROM jewel cases, read me files, packaging materials, quick start guides, workbooks, and marketing literature such as booklets, brochures, catalogs and data sheets.

### ***Review Protocols***

Describe the kinds of reviews planned—**editorial, managerial, and technical**—and who will conduct these reviews. Describe the kinds of usability tests planned and who will perform these tests. Common methodologies are discussed in the section on evaluating online support systems.

### ***Development Schedules***

Include schedules and milestones for all project stages: alpha, beta, and final versions, status and review meetings, editing cycles, usability tests, productions

tasks, debriefings, and maintenance activities. List all major tasks and their completion dates. Note any critical timing issues and dependencies. Build in slip time and flexibility.

### ***Cost Estimates***

Supply project cost estimates. Consider salaries, the amount of work required to produce each screen of information, non-development tasks related to project management, and other resources such contractors, additional training, computer technologies, and research materials.

### ***Dependencies and Risks***

Note all issues that can change schedules and plans and otherwise affect your development processes in unproductive ways. Interview all team members for their concerns. Common dependencies include access to subject-matter experts, accurate and timely reviews, and open lines of communication between team members. Common risks include unstable products, feature changes, and inadequate or unavailable peripheral services for production.

### ***Sign-Off Sheets***

Encourage responsibility and accountability by asking team members to approve their tasks during the development of deliverables. Include signature lines, date lines, and spaces for them to note any concerns.

When distributing specifications, be sure to clarify the role of your readers in a cover memorandum. Ask them to provide productive criticism at this early stage, and be sure to specify what actions you need of them, such as returning changes and comments by a particular deadline.

## ***Designing Online Support Systems***

If developers have not planned well, it's likely that they will encounter disastrous results when designing their online support systems. Surprisingly, many companies have yet to formalize their specification processes and still approach design problems in ad hoc fashions. Moreover, design practices, whether they involve building car engines, light switches, door knobs, or online support, are very complex activities and often require numerous trial-and-error experiences before central goals are met. That is, designers of online support systems must be prepared for ill-defined problem spaces in which meeting the needs of users may be undermined by organizational and other social values, and in which design solutions must be constructed and argued for throughout the development process. Such "wicked" problems—problems having many different possible approaches and solutions—are complicated by a wide range of factors: organizational goals, the task complexity that designers are trying to support, user expectations, hardware and software constraints, the existence of competitive products, and the lack or abundance of available research on particular design features, just to name a few.

### **The Specification as Springboard**

Once a comprehensive specification has been developed, various new constraints and opportunities will immediately demand attention. You may quickly realize that effective design practices require a robust authoring tool, a multidisciplinary design team, constant storyboarding or prototyping (despite careful planning), iterative usability testing, user-centered design perspectives, and direct manipulation features coupled with object-oriented interfaces.

Initially, you must contend with the vast number of available authoring tools that will claim to solve all your complex design problems in elegant ways. Each of these tools, of course, has its advantages and disadvantages: for example, RoboHelp allows

you to link documents in relatively easy ways, HyperCard has a very intuitive scripting language, and WinHelp assumes a good knowledge of Microsoft Word. Ultimately, the features and constraints of your authoring environments will have both productive and unproductive implications for the design of your support system. For instance, in HyperCard 1.0, it was impossible to create “sticky buttons,” links attached to objects (like words) that stick to those objects in useful ways (even if you scrolled those words off screen). Thus, designers working with this system either had to accept such tool-based design constraints or find creative ways of working around them.

During the design process, you will also quickly realize that no single individual has the wide range of skills needed to effectively produce online support systems [Mehlenbacher, 1995]. Multimedia authors, for example, must blend many different skills, including those typically associated with instructional designers, programmers, architects, cartographers, and rhetoricians [Selber, Forthcoming]. Although in every relevant skill area expert advice may not be available or even possible, there may be places in which such advice is absolutely necessary.

Despite the challenges of selecting authoring tools and forming effective teams, most initial design efforts focus on creating a working storyboard. Designers hold meetings and negotiate over intended audiences, the contexts in which a system will be used, the basic functionality of that system, and its ultimate “look and feel” from a user-interface perspective. Notably, companies such as Wang (in the past) and SAS Institute (currently) often include real users in this initial storyboarding process. Unfortunately, many designers still create product plans without consulting either users or potential competition. And, only very recently has storyboarding necessarily included principles for effective interface design; instead, storyboarding has historically emphasized a system’s intended functionality rather than

integrating that functionality, interface design, and task-orientation into an overall design process.

Designers should consider the short- and long-term needs of their audiences during all development tasks. Although there's a strong industry movement to integrate user-centered design practices and usability testing into the overall design process, research suggests that establishing design quality control remains a difficult practical problem [Benson, 1991; Skelton, 1992]. Simply defining quality as it relates to planning, designing, and evaluating is a substantial challenge, particularly given the exponential growth and change currently facing developers of online support systems.

Finally, many corporations have invested in direct manipulation interfaces and Object-Oriented Programming (OOPs). The "drag and drop" philosophy towards iconic objects has been adopted, for example, by Microsoft and other large vending corporations, and [Shneiderman's, 1987] commitment to direct manipulation has finally been instantiated in contemporary software products. Designers should investigate these computing paradigm shifts before developing online support systems, considering which ones might become programming standards for future software releases.

### Global Design Issues

It is critical to consider global issues early in the design process. These issues cannot be ignored because it's often difficult, if not impossible, to incorporate them after the development process has begun. Global design issues that merit serious attention include the following:

- Accessibility
- Maintainability
- Support on Support
- Organization
- Metaphors and Maps
- User control
- Modelessness
- Consistency
- Reversibility and Error Recovery
- Visual Aesthetics

- Navigation
- Feedback Structures and Context Sensitivity

### ***Accessibility***

Online support systems should be easy to access and exit; ideally, users should return to the state in their primary application that they left when seeking assistance. With the advent of large-screen, high-resolution monitors, this design goal is less difficult to achieve. When appropriate, online support can easily be displayed concurrently with primary applications, providing users with the critical ability to keep their original problem contexts in mind while searching for solutions.

### **Maintainability**

Object-oriented approaches to design allow individuals to view codes and scripts as re-usable chunks of information that can be copied, extended, and refined, and that can then be shared or protected, hidden or revealed, to multiple designers. This economical perspective allows designers to create high levels of system consistency, integrity, and updateability that are difficult to achieve following more traditional, functional models of programming.

### ***Support on Support***

Although online support systems should be easy to use without additional layers of support on support, this goal is frequently difficult to attain. The popular Unix-based Emacs editor, for example, asks users to press <^\_> to access help, yet the majority of users don't understand that <^> represents the control key (which may or may not be labeled "Control" on the keyboard). In addition, users may be dismayed to read four to eight help on help panels before accessing actual information for immediate

problem solving. Support on support must be as simple and elegant to use as the support system itself. If it's not, a rigorous review of the system is probably needed.

### ***Organization***

Despite the claims of many hypertext and world-wide web designers, users of support systems often require some form of hierarchical representation and do not feel comfortable “browsing” or “surfing” for the information that they need. In short, users of support are goal-driven: they want to learn applications; they want to cut-and-paste paragraphs; they want to discover which keystrokes can shorten their procedures. No support user works completely or even primarily in “non-linear” ways. While they may branch from one topic to another for immediate benefits, they normally refuse to browse for extended periods of time to find needed assistance.

### ***Metaphors and Maps***

A thorough task analysis during the planning stage is the best way to perceive how users understand online support systems and how those systems will help them accomplish work. Capturing the task expertise of users and their expectations in terms of language, definitions, elaborations, and exceptions to given rules can help designers anticipate mismatches between user expectations and software functionalities. Designers might, for example, begin to question the “friendliness” of terms such as file, window, and disk, if they closely examined their users' perceptions of those features and their functions [Carroll, 1990]. With the Macintosh interface, for instance, we are told that users engage a “virtual” desktop, but few desktops have trashcans on them and most come with telephones and fax machines these days. Generally, well established metaphors and maps help users identify tools in relation to each other, their current states in systems, and, if designed well, the

various paths or selection options available [Winn, 1990]. Importantly, interface metaphors and maps are not pedagogically or even politically neutral: they embody particular ways of knowing and operating in online information spaces [Selber, 1995; Selfe and Selfe, 1994].

### ***Navigation***

It's useful to examine the differences and similarities between terms that describe traditional methods of navigation. Each term—jumping, traversing, searching, browsing, opening, paging, scrolling—reveals various assumptions underlying a designer's perceptions of users, systems, and the user-system interactions involved in navigation. Jumping and traversing, for example, emphasize a system's capabilities or structures. In a hierarchically-organized database structure (common to many contemporary online information systems), users move down through various levels which are usually more specific categorically. For example, in a complex system about restaurants, the top level might provide users with information about restaurant types. One effective organization, often referred to as functional grouping [Mehlenbacher, Duffy, and Palmer, 1989], would list Chinese, American, Irish, German, Canadian, Japanese, and so on. On selecting American, users would move to a submenu that might list Breakfast, Lunch, Dinner, Salads, and Soups. One more level into the hierarchy might list specific food types. Another jump down might reveal short descriptions of different dishes within the previously selected categories of ethnicity and meal type.

Similarly, opening, paging and scrolling are generally system-specific terms for describing navigation through online information spaces. Opening derives its roots from the tasks of opening files, subfiles, and more subsubfiles in hierarchical arrangements. Paging suggests that online information is sequentially organized in discrete chunks that can be viewed one page at a time. Scrolling connotes a linear,

seamless movement up and down (as with ancient scrolls). To clarify assumptions about underlying structure, consider the restaurant system described earlier. Having traversed the hierarchical structure to the list of food descriptions, users might have an opportunity to view a number of relevant meal descriptions. If there are several items, designers will either have to develop another dimension for hierarchical organization (such as spicy and mild) or switch to another organizational model (such as paging). In addition, based on needs assessments, designers must decide if users will need to jump across the hierarchy to see related dishes in other ethnic groups or meal types.

### ***User Control***

Users don't appreciate being reminded that they are relatively powerless over the software applications that they use. Hence the common and confident description that users give to reading print-based books rather than browsing online: "New technology is like MTV: temporally splintered, short-term, and anti-meditative; Generation X, therefore, is superficial." We, as teachers and instructional designers, can appreciate this (arguable) position, but we also recognize the complexity and contradiction that new technologies can introduce into human-computer interactions, modes of community, and communicative expectations. We understand, for example, that historically, MS-DOS experts rejected the Macintosh Graphical User Interface (Mac-GUI) as "hiding" the complexity of computing from end users, despite the subsequent contemporary efforts by those same individuals to emulate and delineate their recent systems from the same computing philosophy. Obviously, computer users are also consumers, and consumers respond to or reject technologies based on a variety of factors, including their learnability, usability, long-term competitiveness, and institutional guidelines. Lotus-1-2-3's system, in convincing its potential audience that producing graphics was an integral part of

selling ideas with data and statistical compilations, buried VisiCalc and its certainty that spreadsheet technology ought to look and feel a particular way and perform x-y-z functions.

### ***Modelessness***

Whenever possible, designers should avoid systems or features that make users operate in distinctly new or unfamiliar ways. Heightened expectations about computer interfaces in general encourage users to manipulate objects without necessarily understanding their particular functions. Users expect that they can “click on anything” and, at the same time, re-click something else to re-establish the state that they only moments ago left. Thus, when users encounter a dialog box providing three options (OK, Save, and Cancel), they often expect that selecting the previous window or state will return them to a situation that they remember, understand, and want to work with based on new information (that they’ve gained from the recently displayed dialog box). Thus, systems like Microsoft Office 6.0 now allow individuals to use the capabilities of many different applications—a spreadsheet, graphics package, and word-processing program—from within the workspace of a single document.

### ***Consistency***

Online support should be consistent from one screen to another, and across the various features that inhabit those screens: for example, text-boxes, headings, control buttons, menu-item names, dialog boxes, and so on. Designers should set parameters early in the design process that establish rules for how all interface features will look and operate as individuals employ support functions. If users are encouraged to develop mental models of how support systems operate, the implications of those models should be consistent across all areas. For example, if

pressing the Help key displays Balloon Help in the main workspace, that key should also display Balloon Help when working in dialogue boxes and in error message windows.

### ***Reversibility and Error Recovery***

Many users freely experiment with online applications. While such experimentation has important pedagogical value, it can have terrible implications for support system designers. Therefore, it's critical to build "undo" or "backup" functions into all systems. Users gain confidence and take learning risks knowing that they can back-track from any mistakes they've made using unfamiliar applications and support. Such user expectations are not impossible to meet if designers provide ways for users to reverse their procedures and easily recover from errors. These features should be accessible at all levels of support system use.

### ***Visual Aesthetics***

One crucial design goal is to produce what's sometimes called "subjective user satisfaction." Researchers generally have a difficult time quantifying why some users prefer Windows 95 over the Macintosh OS over the Unix Motif interface. Individuals who are comfortable using a program in one platform may not want related online support to adopt the visual aesthetics of another. This approach is common on cross-platform applications to both standardize and speed product development, but the negative subjective experience of users may defeat those benefits. Critics of Microsoft Word 6.0 for the Macintosh, for example, frequently complain that its interface mimics that of Windows.

### ***Feedback Structures and Context Sensitivity***

Designers should create system and error messages that are meaningful to people. Such user-centered practices derive from carefully assessing the backgrounds, goals, and work contexts of users to determine the particular support forms and spaces in which assistance will be provided. The ability to move over interface objects and get feedback on their functions, or to click on objects and get useful, related explanations about those objects, is extremely valuable but still relatively rare. Context sensitivity can helpfully shorten feedback loops, allowing users to skip traditional but sometimes distracting methods of finding information about on-screen objects (such as indexes and tables of contents).

### **Local Design Issues**

Local design issues, though less critical to the overall success of online support systems, are nevertheless important to consider. Research suggests that well designed systems are sometimes still criticized if users find spelling errors or screens that are cluttered and difficult to read, or if users dislike the ethos or writing style of the online support system designers [Wright, 1980]. At a minimum, therefore, the local design issues that merit serious attention include the following:

- Levels of Explanation
- Chunking
- Discourse Cues
- Structured Headings
- Bulleting and Lists
- Iconic Markers
- Typographic Legibility
- Negative Space

### ***Levels of Explanation***

Different contexts and purposes require different levels of explanation. Highly focused online help users typically want very concise, even telegraphic information. Elaborations and explanations are more suited to tutorial users, who must build robust mental models to guide subsequent work. For either type of user, however,

an inappropriate level of explanation will certainly impede their performance. In almost every case, users familiar with other forms of technical communication—reports, proposals, and memos, for example—expect simple, concise, active sentence and paragraph structures rather than long, digressing passages (although there are notable exceptions, such as the longer narratives often used in tutorials).

### ***Chunking***

According to Miller [1956], users access information in chunks of seven plus or minus two, and research in document design has extended this finding to the ways users access, process, and retain new information. Therefore, spend adequate design time parsing out information packages that users can consistently access when moving from screen to screen. Chunking can also aid consistency by helping designers separate different kinds of information, in small units, that can be consistently placed across a system. The chunk sizes of information for screens tend to be smaller than those for printed pages because of the numerous physical page/screen issues discussed in the Underlying Principles section. Also, however, chunk sizes vary according to many different rhetorical issues, including user expertise. For example, for an expert Unix user, the command `<chmod go+r *.html>` may be a single chunk of information. For a novice Unix user, that same command should probably be chunked at the character level, resulting in many more pieces of information.

### ***Discourse Cues***

When users of print-based texts read, they rely on certain linear constructs or “grammar[s] of the page” [Selfe, 1989; see also Bernhardt, 1993]. Importantly, these grammars are well established historically: bookmarks, indexes, tables of contents, headers, footers, dog-eared or tabbed pages, and visual indications of reading depth.

Online support systems, however, are foreign ground to print-trained readers—they undermine many well learned access methods by introducing new concepts and reading actions: for example, keyword searching, Artificial Intelligence support systems, Electronic Intelligent Support Systems (EISSs), and Wizard-based learning environments. These systems, while being extensions of hardcopy-based tools in many ways, can complicate the expertise and strategies of print-trained readers.

Designers of online support systems, therefore, should provide numerous discourse cues that anchor users in information that they've already processed and that they'll process in the future [Charney, 1987]. When reading linear texts, it's important for readers to predict their experiences and their texts' realities from one paragraph to the next, from one chapter to the next, or from one frame to the next. That is, even in non-linear situations, users naturally build connections between what they've just processed and what they're currently processing and, ultimately, what they're expecting to process in the very near future.

### ***Structured Headings***

Miller's [1956] argument that chunks of seven items plus or minus two are optimal for processing information is very applicable to this particular design guideline. By definition, headings are concise descriptions of the content that they precede. But, concision in headings is a mixed goal: two-word headings and ten-word headings are least useful to users. Designers should search for some middle ground between concision and verbosity when labeling the information that follows. Studies also suggest that headings should be consistently placed on all screens, and that they should explicitly "spell out" what users can expect in the section that follows [Felker, 1980; Felker, Pickering, Charrow, Holland, and Redish, 1981].

### ***Bulleting and Lists***

Bulleted and numbered lists provide ways of relating large or complicated bodies of information [Carliner, 1987]. These structures distill long or complex concepts that, in traditional sentence form, might appear repetitive or confusing. In addition, in some cases lists can be used as **advance organizers** for sections used in a module or for summaries of information already presented. Bulleted lists should be used for items at the same conceptual level in which ordering is unimportant, while numbered lists should be used when ordering is critical (for example, in task-oriented or rank-ordered information).

### ***Iconic Markers***

Iconic markers are visual items that highlight important elements (warnings and cautions) or help maintain consistency across different screens in the same class. Such graphical elements help readers skim texts for specific types of information. In this way, iconic markers can help designers create a single system that can assist readers with differing expertise levels, goals, and contexts [Horton, 1991; Tufte, 1990].

### ***Typographic Legibility***

The legibility of online type can drastically affect the performance of users. Because readers distinguish letters based on their differences from other letters, “serif” fonts are generally more readable than “sans-serif” fonts. (Serif fonts, such as Times and Palatino, contain ticks or extensions at the ends of letter-strokes, while sans-serif fonts such as Helvetica and Futura have clean ends.) In some contexts, limitations in screen resolution, screen size, or the characteristics of a particular font contradict this guideline, so designers must think carefully about type styles. For example, low-resolution screens sometimes make serif fonts illegible. Notably, as reading from high-resolution monitors becomes more like reading from printed pages, guidelines

for presenting text online will better emulate those relating to hardcopy documents [Haas and Hayes, 1987]. Moreover, because readers process words, in part, by their distinctive shapes, the wider variances of mixed-case words are usually more legible than all-caps words.

In addition, for contrast or for short amounts of text, designers can employ less legible type families in online environments. Headings in a sans-serif font contrast usefully with body text in a serif font; this contrast helps users distinguish among different classes of online text. Finally, the relative degree of openness (or negative space) inside individual letters of fonts can effect legibility. The lower-case letter “a” in Palatino holds more negative space than the same letter in Times, making Palatino more readable in some contexts (such as reading with low-resolution monitors).

### ***Negative Space***

Negative space in online support systems relates to those screen areas not containing text. Although it might seem wasteful to provide adequate negative space, leaving such space is important for at least two reasons: it helps readers see online elements as discrete and separate; and it helps designers limit the amount of information (and number of concepts) they provide on any single screen. Notably, if adequate negative space is provided, only about 1/3 of a printed page will fit on a single screen.

### ***Evaluating Online Support Systems***

Evaluating usability should be a central component of online support system development. Usability refers to the degree of user effectiveness and efficiency in operating online support to complete tasks and achieve goals. Although measuring user effectiveness and efficiency is the core goal in usability testing, it's also useful

for checking the accuracy, consistency, and completeness of included information. From our perspective, usable systems are accessible, maintainable, visually consistent, comprehensive, accurate, and oriented around the tasks that users must perform [Mehlenbacher, 1993].

Historically, the task of evaluating usability has occurred either after online support systems were already shipped or in very late stages of product development (beta or post-beta). This kind of **summative evaluation**, aimed at determining the overall quality of systems and their contents, is very useful in a number of ways: for determining the ability of users to accomplish their targeted tasks and goals using well-formed products, for methodically collecting user feedback for subsequent version releases, and for comparing the features and designs of finished online support systems with those of competing products.

More recent approaches to evaluation, however, expand this rather limited and late-stage approach [Carroll and Rosson, 1985]. It's now quite common for developers to intersperse their planning and designing stages with **formative evaluations**—evaluations that contribute in direct ways to the developments and uses of online support systems. Viewing evaluation as an integral part of system construction has many important advantages: it enlists real users or user surrogates, thereby adding external (customer-centered) perspectives to otherwise internal (developer-centered) ways of knowing; it helps determine what kinds of major revisions and adjustments might be accomplished prior to significant work investments; and it discourages premature closure on system designs until those designs are clarified and refined in iterative ways by members of a user base.

Evaluating the usability of online support systems, either through summative or formative techniques, occurs in both laboratory (controlled) and workplace (natural) contexts, as well through other research techniques employed at greater distances.

Laboratory evaluations allow you to isolate, observe, and track user task performances in systematic ways. These evaluations primarily provide quantitative feedback measured along predetermined scales, although attitudinal and other types of qualitative information can be collected. Because special equipment is required in these research environments—a physical space, video camera, audio recorder, and/or one-way mirror, for instance—developers with limited usability resources may rely solely on workplace evaluations and other data collection techniques.

Workplace evaluations take place in the normal environments of users. As such, these evaluations allow developers to examine how well their online support systems are employed and understood within the day-to-day contexts of users, their tasks, and their organizations. Notably, workplace evaluations are still informed by systematic approaches to research. In fact, the qualitative feedback gathered in these instances is commonly informed by empirical designs [Bjerknes, Ehn, and Kyng, 1987; Blomberg and Henderson, 1990; Piela, McKelvey, and Mehlenbacher, 1991]. Because workplace evaluations occur at customer sites, they require special access and a high degree of trust between users and developers [Brown and Duguid, 1992].

Other research techniques allow developers to supplement usability data collected from laboratory and workplace contexts. They can gather both quantitative and qualitative feedback by conducting telephone interviews, administering surveys, compiling literature reviews, and reading related studies. Data collected from all contexts, when examined together, can provide useful directions for both system developments and revisions [MacNealy, 1992].

Importantly, the research contexts that developers select will influence their usability results and conclusions in central ways. Experimental studies conducted in controlled laboratories, for example, may yield task completion rates that are either unrealistic or unmeaningful within normal working environments. For this

reason, triangulate research contexts whenever possible. Also, the contexts that developers select should support and not drive their research goals, which, in turn, help them articulate the appropriate forms of evaluation needed for testing usability.

### Forms of Evaluation

There are many different forms of evaluation available to developers of online support systems. We briefly describe nine forms that might be employed during early-, middle-, and late-phase efforts. These forms are summarized in Table 4. Although many other techniques exist for testing usability, these represent common, industry-standard approaches. For comprehensive discussions of these and other forms, see [Goubil-Gambrell, 1992; Nielsen, 1993; Dumas and Redish, 1993].

#### ***Early-Phase Evaluations***

Focus groups, task analyses, and prototype walkthroughs are useful forms of evaluation during preplanning, planning, and developmental stages. Focus groups help developers explore the concerns, beliefs, and preferences of real users, as well as understand their perspectives on an organization and its existing products and practices (if any). During focus group sessions, which usually last between one and three hours, five to 10 users respond to a series of pre-planned questions formulated by experienced researchers. These sessions are best lead by individuals who have experience with group dynamics and with focusing and re-focusing group discussions in both efficient and productive ways.

As discussed in the planning section, task analyses identify the complex range of actions that users must perform, and they further divide and subdivide these actions into subactions that the planned online support systems should document and discuss. In these analyses, developers both interview and observe users to

understand their typical ways of working and accomplishing tasks. From this information, they extract the working goals of users, the system features and other ancillary information—both print-based and online—that support these goals, and the full procedures that users follow to accomplish all tasks.

Prototype walkthroughs provide early opportunities in which users can evaluate the planned contents and designs of online support systems [Dillard, 1992]. As the name of this technique suggests, users explore working models of systems and attempt to accomplish tasks using a limited number of features. From data collected by observing and interviewing users, developers proceed with more stable and aggressive instantiations. Prototype walkthroughs allow designers to collect user feedback prior to investing substantial resources in system development.

### ***Middle-Phase Evaluations***

Protocol analyses, performance tests, and contextual inquiries are useful forms of evaluation during pre-beta and beta stages. Protocol analyses ask individuals to verbalize their thoughts while using online support systems and/or after they have finished. These thoughts, which are recorded on audiotape and or/videotape, help reveal both confusing and missing information [Lewis, 1982]. They also reveal the kinds of expectations that users typically have. Employing this approach, developers target a limited number of representative users and collect richly textured insights into their normally hidden thought processes.

Performance tests measure how well users accomplish a pre-determined set of tasks measured along pre-determined scales [Liebried and McNair, 1992]. Users are tested either in laboratory or workplace contexts, and the data collected from these evaluations is quantitative and/or qualitative. During performance tests, it's important to clearly delineate where user tasks begin and where they end. Common measurements during performance tests include the amount of time taken to

complete particular tasks, the number of errors made during task performances, and the most and least frequently used features in a system.

Contextual inquiries examine individual users and how they accomplish tasks in their own working environments [Anderson, 1989]. Through observations and interviews, developers construct models of user behavior in real-world settings, determining the ways in which their online systems should support those behaviors. Obviously, this techniques is also extremely useful during developmental phases.

### ***Late-Phase Evaluations***

User-advocate reviews, competitive analyses, and user surveys are useful forms of evaluation during post-beta stages and after online support systems have been distributed to customers. As opposed to enlisting real users, user-advocate reviews ask usability specialists to assess the merits and demerits of online support systems. Like real users, these specialists can perform a wide range of pre-planned tasks identified by developers. Unlike real users, however, they pay particular attention to how well online support systems reflect sound human-computer interaction principles.

Competitive analyses examine the strengths, weakness, and overall designs of competing products, helping developers differentiate their systems from others in the marketplace. Such analyses often occur in laboratory settings and measure system performances under a wide variety of conditions and with a wide range of hardware and software configurations.

And, user surveys help developers both understand customer requirements and systematically collect their feedback. Surveys are generally inexpensive to administer and can provide users with anonymous avenues for providing honest

commentary. Importantly, they require researchers who are skilled at developing both valid and useful lines of inquiry which are both targeted and open-ended. User surveys can be administered by mail, electronic mail, over the telephone, or in face-to-face environments [Zimmerman and Muraski, 1995].

### Procedures for Evaluation

In general, there are eight sequential steps in evaluating online support systems:

1. Forming an evaluation team
2. Identifying evaluation goals
3. Selecting evaluation methods
4. Developing realistic scenarios
5. Enlisting real users
6. Implementing the evaluation
7. Analyzing the data
8. Distributing the findings

1. Form an evaluation team. Include researchers experienced in conducting usability tests and in employing empirical methods, developers involved in the project, personnel with access to users, product engineers, and marketing specialists.
2. Identify what you hope to accomplish during the evaluation. Clearly state your usability goals in specific terms, paying particular attention to the concerns of all team members, and to what constitutes success and failure during these evaluations.
3. Select appropriate evaluation methods and contexts for accomplishing your objectives. Identify the methods and contexts that are realistically available to you and your evaluation team. Select those which can best support your usability goals.
4. Develop scenarios and real tasks that can help you achieve your evaluation objectives. Make each scenario or task a discrete module that can be easily measured, observed, and/or discussed. Use both direct and indirect approaches. The former asks users to perform specific actions and tasks, while the latter

provides situations in which users select their actions from within emerging situations. Create a **test checklist** for reference.

5. Enlist real users to participate in the evaluation. Select a range of individuals, including representative users, novices, and experts. The kind and number of users required will depend on the evaluation forms and contexts you select. Be sure to provide real incentives for user participation.
6. Implement your evaluation. Explain the process to participants, build a comfort level for them, get signatures on nondisclosure agreements (if required or appropriate), conduct the evaluation, and debrief the participants. Importantly, there are direct connection between your testing techniques and the usefulness of your results. Work carefully and validly.
7. Evaluate the results of your tests. Examine all data—quantitative and qualitative—that you have collected, and consider the effects of form and context on your results. Interpret the results of your research modestly. Be careful not to generalize from what you learn from one or two users working in their own contexts.
8. Formalize and distribute the results of your evaluation. Share your methodologies, results, conclusions, and recommendations with all team members in a completion report.

## **RESEARCH ISSUES AND SUMMARY**

While planning, designing, and evaluating online support systems, there are many open research issues that can complicate the development practices we've outlined. From our perspective, the most difficult challenges generally relate to interpersonal, cultural, and pedagogical areas, both internally (among development team members) and externally (among customers). Specifically, five open research issues

that remain under-discussed in the literature on online support systems include the following: collaboration, diversity in the workplace, translation and internationalization, user motivation and aesthetic response, and designer training.

### **Collaboration**

As we've already mentioned, no one designer is likely to possess all the diverse skill sets needed to effectively develop online support systems. Interdisciplinary collaborative teams, therefore, are commonly assembled in both small and large organizations, and they normally include many different kinds of specialists.

Although collaboration has become an integral part of online support system development, collaborative activities are not always successful or without their share of complications. As [Selber, McGavin, Klein, and Johnson-Eilola, 1996] note, difficulties and failures among and within work groups are numerous and wide-ranging for a variety of reasons: group leadership may be ineffective; individual agendas may conflict with group agendas; individuals may be unwilling or unable to discuss differences of opinion; unequal power relations may exist among group members due to various social forces such as organizational rank; and the simple logistical issues of coordinating schedules and tasks.

Moreover, the ways in which groups collaborate are increasingly expanding in both time and space, and this expansion raises many new questions about the nature of collaboration. Using computer-supported cooperative work tools, designers can now interact online in both asynchronous and synchronous modes, and such modes, many claim, have the potential to enrich collaborative activities in both predictable and unpredictable ways [see Greenberg, 1991]. Importantly, we would argue that such technological possibilities represent only one factor in fostering productive group work. Changes in organizational structures, group dynamics, and institutional reward systems are also needed for new electronic forms of collaboration to be productively integrated into existing corporate cultures.

### **Diversity in the Workplace**

As workplace diversity increases, online support system designers will be working in, and developing products for, groups not adequately addressed by old ways of communicating, most of which were developed for homogeneously segregated audiences. Traditional conventions and procedures for communicating, while often considered “normal” or “standard,” commonly represent only one (potentially limiting) approach. This diversity issue is not one of political correctness, but of fundamental concerns for productivity and usability. In some cases, relying on traditional uses of language offends colleagues and users: generic uses of “he,” to name but one controversial example, constructs users as explicitly male [see Frank and Treichler, 1989, for a comprehensive discussion of language, gender, and professional writing issues]. In other cases, communication methods and learning styles that work well for certain groups may actually inhibit the learning of others. The differences in high-context cultures such as the United States and low-context cultures such as Japan is one, often-cited example; Japanese readers may find American styles too abrupt, while American users may find Japanese styles overly formal [see Hoft, 1995, for many additional concerns]. General trends of increasing diversity in both the workplace and marketplace will only multiply the need for additional research in this areas.

### **Translation and Internationalization**

The increasing diversity in the workplace noted above relates to translation and internationalization issues in central ways. The growth of global economic systems and marketplaces frequently demands that online support system designers recognize that their work will be used by people in other countries and cultures. Although the research and development of translation tools has assisted designers by automating some portions of the translation process, more work is needed before these systems begin to approach the skills of human translators [Spalink, 1995]. In

addition, while literal translations can sometimes provide partial support for users, broader cultural differences can impede usability to a similar degree. Over the last few decades, researchers have begun to investigate differences in communication styles (direct vs. indirect), learning patterns (visual vs. verbal), and work habits (hierarchical vs. team) inherent in different cultures. While some portions of translation, especially literal translation, can be easily automated or out sourced, cultural differences may reside in the very framework of support systems and their interfaces. Issues of translation and internationalization, therefore, are critical for designers to consider.

### **User Motivation and Aesthetic Response**

Capturing the particular emotive reactions of users to online support systems is an important research area that has received limited attention. Less cognitive than affective, subjective user reactions to human-computer interfaces and to the content of online support systems can encourage individuals, for example, to embrace one particular applications (such as Windows 3.1) over another (such as Windows 95), even though the application that they reject may be superior in form and function. Still, with the exception of [Shneiderman, 1987], who has built a useful but limited likert-scale test aimed at uncovering subjective user reactions to software, few researchers have emphasized the role of emotion and affect in human-computer interaction. Moreover, it's clear that researchers are not likely to study what their research methods can't detect, so various efforts are currently in place to re-examine and re-evaluate the methods employed to gauge user satisfaction. Researchers have begun to contrast and compare more traditional quantitative methods with emerging ethnographic methods [MacNealy, 1992; Sullivan and Spilka, 1992], and to explore how people work apart from how they work with our tools [Bannon, 1995]. As we better understand user preferences, inclinations, and generalizable patterns

across cultural, political, and educational levels, user motivation and aesthetic response will certainly gain more attention in online support system development.

### **Developer Training**

Because developing online support systems is a relatively new type of computer-based work, many degree programs and professional development seminars are rightly tentative in claiming one absolute curriculum that's appropriate for this educational area. In technical communication programs alone, for example, there are different courses of study for online support system designers in English departments, humanities departments, communication and rhetoric departments, and technical journalism departments. This list, of course, does not include those many programs in departments outside the disciplines traditionally interested in language studies: computer science and engineering, human factors, instructional design, psychology, and so on. To complicate curricular matters further, each of these programs privileges particular theoretical perspectives and research methodologies that guide the ways in which they examine human-computer interaction problems. We would argue, however, that such diversity is a strength rather than weakness given the complexity of most design activities.

Although the orientation of any particular program should reflect the strengths of its faculty and facilities, in general curricula for preparing online support system designers should be interdisciplinary and balance production, literacy, and humanistic concerns [Selber, 1994]. Even ostensibly simple design tasks, such as selecting background colors or shaping icons, require individuals to blend research findings and technical competencies from many different fields: in this case, visual design, psychology, and rhetoric. This example, in addition, highlights the need for programs to balance the kinds of issues they centrally discuss. A focus on production processes would help designers learn the mechanics of these two tasks. Further

discussions of literacy, however, might reveal that certain colors connote particular meanings in various cultural contexts or that cultural diversity complicates icon shapes commonly perceived as “standard.” Even further considerations of humanistic concerns might demonstrate the ways in which icons (and other visual elements) can be unproductively gendered. Without this kind of interdisciplinary work and balanced training, designers will be unprepared to make the kinds of rhetorical judgments outlined in this chapter.

### **Summary**

The three development areas we’ve discussed—planning, designing, and testing—require designers to consider rhetorical as well as technical issues, recursive rather than linear approaches, and qualitative as well as quantitative methods for understanding user behavior. Our increasingly sharper focus on the human dimensions of online support highlights the complexity of productively understanding human-computer interactions within the richly textured cultural contexts of users and their work.

Underlying principles encourage designers to consider the advantages and disadvantages of online support (paper-based solutions may still be appropriate in certain cases); the physical and rhetorical differences between print-based and online support, particularly in terms of organizational, navigational, and contextual structures; and rhetorical frameworks for online support: namely, users, their goals, and their time/space frames.

Best practices can be divided into planning, designing, and testing phases, with each of these phases potentially affecting the other two. At a minimum, planning requires designers to conduct needs assessments and develop specifications, two early activities essential to meeting both external (user) and internal (designer/manager) needs. Designing encourages individuals to consider both global and local issues, and to gauge the success of their strategies in terms of user

satisfaction. Evaluating includes both formative and summative usability tests, and provides a framework for such testing during early, middle, and late phases of development.

As the technological aspects of online support systems and authoring applications continue to evolve, it will be increasingly important for designers to consider the rhetorical dimensions of their work. The best solutions for users will not come solely from understanding technological innovations but from support system environments designed with a wide range of technological, organizational, cultural, and instructional factors in mind.

## **DEFINING TERMS**

**Advance organizers:** Overview statements, often in bulleted or list form, that provide a summary of what some textual unit (an entire online support system, module, or submodule) will contain. They also often discuss how that unit is organized and how it might be used given users with differing knowledge bases and goals.

**Chunking:** A design stage in which developers divide the content for their online support systems into discrete, easily identifiable units.

**Documentation:** A type of online support system that provides reference material or discusses extensive procedures. It has a narrower pedagogical scope than tutorials but a broader one than help.

**Editorial review:** A development stage, usually late in the process, in which online support systems are examined for consistency, style, mechanics, and completeness.

**Formative evaluations:** A type of evaluation occurring while online support systems are being developed. In general, they provide feedback that can be used to reconceive or revise systems before they are finished.

**Help:** A type of online support system that provides brief information for solving particularly pressing problems. This kind of support often includes context-sensitivity. It has a narrower pedagogical scope than both tutorials and documentation.

**Managerial review:** A development stage, often occurring many different times during the life of a project, in which online support systems are examined for business-related concerns (project costs, customer requests, and so on).

**Online support systems:** Digital forms of assistance—namely, tutorials, documentation, and help—that aid computer-based learning and task-oriented activities.

**Primary work:** Job-related tasks, such as writing a newsletter, managing a budget, or learning a procedure, for which individuals rely on computer technologies for their successful completion.

**Summative evaluations:** A type of evaluation occurring after online support systems are finished or in very late stages of development. In general, they measure the success of final design decisions.

**Technical review:** A development stage, often occurring many different times during the life of a project, in which online support systems are examined for performance issues and by subject-matter experts.

**Test checklist:** An outline of usability test procedures. It helps remind usability testers of their planned procedures and the proper order of these procedures.

**Tutorials:** A type of online support system that provides a supportive and safe educational environment for learning processes or products. They usually include interactivity, and the content and pace are controlled, to some degree, by users. Tutorials have a broader pedagogical scope than both documentation and help.

**User-centered design:** Development practices that privilege rhetorical and humanistic considerations, such as users, their goals, and their social contexts, over technological considerations, such as system capabilities.

## REFERENCES

- Anderson, R. 1989. Notes about some experiences with contextual research. *SIGCHI Bulletin*. 20 (4): 29-30.
- Bannon, L. J. 1995. The Politics of Design—Representing Work. *Communications of the ACM*. 38 (9): 66-68.
- Beabes, M. A. and Flanders, A. 1995. Experiences with using contextual inquiry to design information. *Technical Communication*. 42 (3): 409-420.
- Benson, T. 1991. Challenging global myths: International quality study. *Industry Week*. 240 (19): 12-25.
- Bernhardt, S. 1993. The shape of text to come: The texture of print on screens. *College Composition and Communication*. 44 (2): 151-175.
- Bjerknes, G., Ehn, P., and Kyng, M. 1987. *Computers and Democracy*. Avebury: London, England.
- Blomberg, J. and Henderson, A. 1990. Reflections on participatory design: Lessons from the Trillium Experience. In *Proceedings of CHI'90: Human Factors in Computing Systems*, p. 353-360. ACM, Seattle, WA.
- Boy, G. 1992. Computer integrated documentation. In *Sociomedia*, ed. E. Barrett, p. 507-532. MIT P, Cambridge.
- Brown, J. S. and Duguid, P. 1992. Enacting design for the workplace. In *Usability: Turning Technologies into Tools*, ed. P. S. Adler and T. A. Winograd, p. 164-197. Oxford University Press, New York, NY.
- Caernarven-Smith, P. 1990. Annotated bibliography on costs, productivity, quality, and profitability in technical publishing: 1956-1988. *Technical Communication*. 37 (2): 116-121.
- Card, S. K., Moran, T. P., and Newell, A. 1983. *The Psychology of Human-Computer Interaction*. Lawrence Erlbaum, Hillsdale, NJ.
- Carliner, S. 1987. Lists: The ultimate organizer for engineering writing. In *Writing and Speaking in the Technology Professions: A Practical Guide*, ed. D. Beer, p. 53-56. IEEE Press, New York, NY.
- Carroll, J. 1990. *The Nurnberg Funnel*. MIT Press, Cambridge, MA.
- Carroll, J. M. and Rosson, M. B. 1985. Usability specifications as a tool in iterative development. In *Advances in Human-Computer Interaction*, ed. H. R. Hartson, p. 1-28. Ablex, Norwood, NJ.
- Charney, D. 1987. Comprehending non-linear text: The role of discourse cues and reading strategies. In *Hypertext '87 Papers*, p. 109-120. ACM, Chapel Hill, NC.
- Dillard, J. D. 1992. Maximizing documentation usability and product quality through structured rapid prototyping. In *Proceedings of the 38<sup>th</sup> International Technical Communication Conference*, p. 119-122. Society for Technical Communication, Arlington, VA.

- Duffy, T. M., Palmer, J. E., and Mehlenbacher, B. 1992. *Online Help: Design and Evaluation*. Ablex, Norwood, NJ.
- Dumas, J. and Redish, J. A. 1994. *A Practical Guide to Usability Testing*. Ablex, Norwood, NJ.
- Felker, D. 1980. *Document design: A review of the relevant research*. American Institutes for Research, Washington, DC.
- Felker, D., Pickering, F., Charrow, V., Holland, V., and Redish, J. 1981. *Guidelines for Document Designers*. American Institutes for Research, Washington, DC.
- Frank, F. W., and Treichler, P. *Language, Gender, and Professional Writing*. 1989. MLA, New York, NY.
- Goubil-Gambrell, P. 1992. A practitioner's guide to research methods. *Technical Communication*. 39 (4): 582-591.
- Greenberg, S., ed. 1991. *Computer-supported Cooperative Work and Groupware*. Academic Press, New York, NY.
- Haas, C. 1989. Seeing it on the screen isn't really seeing it: Computer writers' reading problems. In *Critical Perspectives on Computers and Composition Instruction*, ed. G. Hawisher and C. Selfe, p. 16-29. Teacher's College Press, New York, NY.
- Haas, C., and Hayes, J. R. 1987. *Effects of text display variables on reading tasks: Computer screen vs. hard copy*. ERIC No. ED 260 387. Communications Design Center, Carnegie Mellon University, Pittsburgh, PA.
- Hoft, N. 1995. *International Technical Communication: How to Export Information about High Technology*. John Wiley, New York, NY.
- Horton, W. 1991. *Illustrating Computer Documentation: The Art of Presenting Information Graphically on Paper and Online*. John Wiley, New York, NY.
- Johnson-Eilola, J. In press. *Nostalgic Angels: Rearticulating Hypertext Writing*. Ablex, Norwood, NJ.
- Lewis, C. 1982. Using the "thinking-aloud" method in cognitive interface design. In *Research Report-9265-40713*, p. 1-6. IBM Thomas J. Watson Research Center, Yorktown Heights, NY.
- Liebried, K. H. and McNair, C. J. 1992. *Benchmarking: A Tool for Continuous Improvement*. Harper Collins, New York, NY.
- MacNealy, M. S. 1992. Research in technical communication: A view of the past and a challenge for the future. *Technical Communication*. 39 (4): 533-551.
- Mehlenbacher, B. 1993. Software usability: Choosing appropriate methods for evaluating online systems and documentation. In *SIGDOC 93: The 11<sup>th</sup> Annual International Conference Proceedings*, p. 209-222. ACM, New York, NY.
- Mehlenbacher, B. 1995. Charting the future of technical communication: SIGDOC 94 and the great divide. *The Journal of Computer Documentation*. 19 (2): 15-21.
- Mehlenbacher, B., Duffy, T. M., and Palmer, J. E. 1989. Finding information on a menu: Linking menu organization to the user's goals. *Human-Computer Interaction*. 4 (3): 231-251.
- Miller, G. 1956. The magical number seven, plus or minus two: Some limits on our capacity for processing information. *The Psychological Review*. 63: 81-87.
- Nielsen, J. 1993. *Usability Engineering*. Academic Press, Boston, MA.
- Paradis, James. 1991. Text and action: The operator's manual in context and in court. In Charles Bazerman & James Paradis (Eds.), *Textual dynamics of the professions:*

- Historical and contemporary studies of writing in professional communities*, ed. C. Bazerman and J. Paradis, p. 256-278. U of Wisconsin P, Madison.
- Petrauskas, B. 1991. Online reference system design and development. In *Perspectives on Software Documentation: Inquiries and Innovations*, ed. T. Barker, p. 243-272. Baywood, Amityville, NY.
- Piela, P., McKelvey, R., and Mehlenbacher, B. 1991. Integrating Usability and participatory design into basic engineering design research. *Engineering Design Research Center Technical Report*. Carnegie Mellon University, Pittsburgh, PA.
- Rubens, P., and Krull, R. 1985. Applications of research on document design to online displays. *Technical Communication*. 37 (4): 29-34.
- Selber, S. 1994. Beyond skill building: Challenges facing technical communication teachers in the computer age. *Technical Communication Quarterly*. 3 (4): 365-390.
- Selber, S. 1995. Metaphorical perspectives on hypertext. *IEEE Transactions on Professional Communication*. 38 (2): 59-67.
- Selber, S. Forthcoming. The politics and practice of media design. In *Theory, Practice, and Program Design in Technical Communication: Foundations for Teaching an Emergent Discipline*, ed. K. Staples and C. Ornatowski. Ablex, Norwood, NJ.
- Selber, S., McGavin, D., Klein, W., and Johnson-Eilola, J. 1996. Issues in Hypertext-Supported Collaborative Writing. In *Nonacademic Writing: Social Theory and Technology*, ed. A. H. Duin and C. Hansen. Lawrence Erlbaum, Hillsdale, NJ.
- Selfe, Cynthia. 1989. Redefining literacy: The multilayered grammars of computers. In *Critical Perspectives on Computers and Composition Instruction*, ed. G. Hawisher and C. Selfe, p. 3-15. Teacher's College Press, New York, NY.
- Selfe, C., and Selfe, J. 1994. The politics of the interface: Power and its exercise in electronic contact zones. *College Composition and Communication*. 45 (4): 480-504.
- Shneiderman, B. 1987. *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, Reading, MA.
- Skelton, T. M. 1992. Testing the usability of usability testing. *Technical Communication*. 39 (3): 343-359.
- Smith, D. 1994. Estimating costs for documentation projects. In *Publications Management: Essays for Professional Communicators*, ed. O. Allen and L. Demming, p. 143-151. Amityville, NY: Baywood.
- Spalink, K. Document Design with Translation in Mind. *Intercom*. 42 (7): 38-43.
- Suchman, L. A. 1987. *Plans and Situated Actions: The Problem of Human-Machine Communication*. Cambridge U P, New York.
- Sullivan, P. and Spilka, R. 1992. Qualitative research in technical communication: Issues of value, identity, and use. *Technical Communication*. 39 (4): 592-606.
- Tufte, E. R. 1990. *Envisioning Information*. Graphics Press, Cheshire, CT.
- Winn, W. 1990. Encoding and retrieval of information in maps and diagrams. *IEEE Transactions on Professional Communication*. 33 (3): 103-107.
- Winograd, T. and F. Flores 1987. *Understanding computers and cognition: A new foundation for design*. Reading, MA: Addison-Wesley.
- Wright, P. 1980. Usability: The criterion for designing written information. In *Processing Visible Language, Vol. 2*, ed. P. A. Koler, M. E. Wrolstad, and H. Bouma, p. 183-206. Plenum Press, New York, NY.

Zimmerman, D., and M. L. Muraski. 1995. *The Elements of Information Gathering: A Guide for Technical Communicators, Scientists, and Engineers*. Oryx, Phoenix, AZ.

## **FURTHER INFORMATION**

In addition to the sources already cited in this chapter, the following books and journals are additional recommended readings. Although the following lists are in no way comprehensive, they represent a starting place for further information about how to plan, design, and evaluate effective and usable online support systems.

### **Recommended Books**

Adler, P. S. and Winograd, T. A., eds. 1992. *Usability: Turning Technologies into Tools*. Oxford University Press, New York, NY.

Barker, T., ed. 1991. *Perspectives on Software Documentation: Inquiries and Innovations*. Baywood, Amityville, NY.

Barnum, C. M. and Carliner, S. 1993. *Techniques for Technical Communicators*. Macmillan, New York, NY.

Barrett, E., ed. 1992. *Sociomedia: Multimedia, Hypermedia, and the Social Construction of Knowledge*. MIT Press, Cambridge, MA.

Brooks, T. 1991. *An Introduction to Human-Computer Interaction*. Lawrence Erlbaum, Hillsdale, NJ.

Brockmann, R. 1992. *Writing Better Computer User Documentation: From Paper to Online*, 2<sup>nd</sup> ed. John Wiley, New York, NY.

Doheny-Farina, S., ed. 1988. *Effective Documentation: What We Have Learned from Research*. MIT Press, Cambridge, MA.

Horton, W. 1990. *Designing and Writing Online Documentation: Help Files to Hypertext*. John Wiley, New York, NY.

Laurel, B., ed. 1990. *The Art of Human-Computer Interface Design*. Addison-Wesley, Reading, MA.

Norman, D. 1988. *The Design of Everyday Things*. Basic, New York, NY.

### **Recommended Journals**

ACM Hypertext

Communications of the ACM

Human-Computer Interaction

IEEE Transactions on Professional Communication

International Journal of Human-Computer Interaction

International Journal of Man-Machine Studies

Journal of Computer-Based Instruction

**SIGCHI Bulletin**

**Technical Communication**

**The Journal of Computer Documentation**

**Table 1. Physical Differences between Print-Based and Online Support**

	<b>Pages</b>	<b>Screens</b>
<b>Resolution</b>	70-1200 dots per inch	50-100 dots per inch
<b>Display Area</b>	generally larger	generally smaller
<b>Aspect Ratio</b>	generally taller than wide	generally wider than tall
<b>Presence</b>	physical static immutable	virtual static dynamic interactive mutable

**Table 2. Rhetorical Differences between Print-Based and Online Support**

	<b>Pages</b>	<b>Screens</b>
<b>Organizational</b>	linear familiar hierarchical logical/deductive fixed	linear and nonlinear familiar and unfamiliar hierarchical and non- hierarchical logical/deductive associative and dynamic
<b>Navigational</b>	familiar limited static	familiar and unfamiliar robust static and dynamic
<b>Contextual</b>	generally rich	generally poor

**Table 3. Rhetorical Frameworks for Online Support**

	<b>Help</b>	<b>Documentation</b>	<b>Tutorials</b>
<b>Users</b>	expert	intermediate	novice
<b>Goals</b>	narrow/ short-term	medium/ short-term	broad/ long-term
<b>Time/Space Frames</b>	parasitic/ internal	parallel/ internal	encompassing/ external