# Notes Toward a Socially Informed Pedagogy for Computer Documentation

Stuart A. Selber
Department of English
Penn State University
University Park, PA 16802
selber@psu.edu

## Abstract

*This article extends Johnson-Eilola's main argument and then, using a thought experiment, examines an extended example of its implications. The experiment follows a student who learns how to produce technical communication artifacts following the philosophy that informs most technical communication classes and that leads to production of the functional but not conceptual systems Johnson-Eilola critiques. The article concludes by recommending changes in overall curricula and in individual courses that would better educate communicators to account for the social implications of their work.*

**H.5.2 User Interfaces**—*training, help, documentation*
**Keywords:** *online help, documentation, user interface*

Johndan Johnson-Eilola makes several significant points about how computer documentation tends to get represented, treated, and understood by those who use it, by those who design it, and by those who teach about how to design it. These points are conceptual rather than practical and their implications challenge us to re-imagine the education of computer documentation specialists in ways that are more attentive to social issues. His argument about the state of affairs in computer documentation hinges on the assertion that the design goal of transparency, while commonsensical and useful on some level, ultimately encourages an impoverished practice and limits our perspective on the nature and purpose of documentation systems. Let me quickly revisit this design goal, which has become axiomatic in functional discussions of design as well as enshrined in landmark essays on usability.

In a general sense, those who promote transparency as a design goal feel it is better to conceal than to reveal the inner workings of an artifact. The assumption is that these are not critical to end users, and in fact can become a barrier to getting the full benefit of what the artifact has to offer. A basic Internet search engine masks its functionality with a barebones interface that only requires users to enter key words or phrases. This interface allows users to focus on the tasks at hand while minimizing what they need to know about how search engines operate. There are obvious benefits here that should not be minimized. On the other hand, the trade-off is that basic Internet searches frequently return results that are too numerous and irrelevant. Yet users who have

over-relied on machine automation for research purposes are often prepared to do little more than change their key words or phrases and try again, as my students often discover. So an implication of transparent design is that it can de-skill computer users by discouraging a focus on broader conceptual knowledge (in this case, knowledge about how to conduct effective secondary research). But there is another implication as well. As Johnson-Eilola argues, the design goal of transparency can discourage a critical awareness of what is happening with and to users in technological environments. These implications should be of interest to those documentation specialists who consider themselves to be user advocates.

It is against this backdrop of concern that Johnson-Eilola encourages us to open up the black box of technology in ways that might benefit computer users, who knowingly or not often struggle with the problem of access to understanding. The solutions he offers are largely social in character: instantiate collaborative models of support that leverage the expertise and generosity of users; help users read interfaces as interested maps that shape both work and the worker; collapse the binary division between procedural and conceptual knowledge, or at least associate these two knowledge types more explicitly in documentation systems; and deal with complexity as an attribute and not a flaw of communication. I consider these suggestions to be a tentative starting point rather than a definitive or exhaustive prescription for action.

That is my general take on the little machines argument, and I want to respond as a sympathetic reader and as a teacher who wonders what might be required in pedagogical terms in order to promote the socially informed vision that Johnson-Eilola lays out. My impression is that most university courses do not fully prepare students to think in expanded ways about computer documentation. I base this claim not only on anecdotal evidence but also on the courses in documentation writing that have been described in the published literature on technical communication. The course I want to review here was developed by Carolyn Boiarsky and Michael Dobberstein (1998) for students at Purdue University Calumet (CUP). I do not single out this course to criticize it, but for the opposite reason: to my mind, it typifies the kind of thoughtful (if traditional) approach one can find in numerous classroom settings at both the undergraduate and graduate level.

Boiarsky and Dobberstein organize their course in a highly structured manner, one that builds up to a major documentation project developed for real users (other CUP students) in situated contexts (the computer labs on campus) and that mirrors standard development processes in the workplace. Students are prepared for the major project through a series of instructional activities that increasingly disclose the complexities involved in creating documentation. The course starts with lessons that teach skills in problem solving, the assumption being that students must first learn how to analyze and research a situation and discuss it meaningfully with their peers. Toward that end, the lessons introduce various heuristics that help students develop strategies and vocabularies that can be mobilized in future problem-solving contexts. The heuristics lead rather naturally to discussions of audience and task analysis, and the course considers these aspects in a document-specific fashion. That is, these lessons focus on the particular needs and concerns of documentation users, which relate to such things as work tasks, computer knowledge, learning preferences, and habits of contending with technological impasses. The next lessons that are scaffolded into the course take up the conventions of the genre. In this segment, students learn what documentation typically contains, how it is typically structured, and how it is typically created and managed. This is a lot to learn and, as Boiarsky and Dobberstein point out, a single textbook or approach is rarely if ever sufficient. Once students are introduced to the conventions of the genre, they learn more about document design and its principles and about the technologies that can support the production of documentation systems. All of the lessons in the course are sequenced from simple to complex and from familiar to unfamiliar, and they are framed by a major project that is authentic in its cognitive demands. As such, the course design draws on important tenets of constructivism.

What can we expect of a student who has completed this type of course, which I would argue is both conventional and useful? A cursory response would list off an array of competencies that can be derived either implicitly or explicitly from the stated course objectives. But this kind of characterization can lack a real sense of clarity for most of us unless it is illustrated in a very concrete manner. So let me attempt to do so with an extended thought experiment that should model the reasoning process of someone who has successfully completed a course like the one Boiarsky and Dobberstein describe. Its purpose is to provide a clearer sense

of the relationship between conventional instruction and a socially informed pedagogical orientation.

The thought experiment takes the form of an analysis of the strengths and weaknesses of SiteInspector, an online utility for testing the usability of Web sites (Figure 1). My feeling is that any student who has taken a course in computer documentation should be able to perform such an analysis for any of the key areas in the syllabus, especially usability. In fact, this is what I assign as a final project in my course because I want to test the extent to which students can employ a meta-discourse to talk about documentation and its practices. Although my exemplar student, Debbie, is a creation of my own, the analysis below emulates the reasoning process of students I have worked with in the recent past.

Before I begin, however, I should briefly explain how SiteInspector works and why I chose this particular artifact for examination. As I mentioned, SiteIn-



**Figure 1: SiteInspector**

spector is an online utility for testing the usability of Web site designs. Designers enter a URL, and SiteInspector examines the associated pages for compatibility, validity, and efficiency issues. More specifically, diagnostic tests are run to determine if the pages display differently across platforms, if there are any broken links, if the HTML is coded correctly, if the site loads quickly, if there are any spelling mistakes, if the site is popular, and if the site is ready to be indexed by directories and search engines. After the tests are run, SiteInspector generates reports that describe usability problems and offers guidelines for fixing them (Figure 2).

I chose SiteInspector because its features reflect an orientation toward Web site design and usability that students of computer documentation should be able to put into proper perspective. The orientation of SiteInspector could be considered technical in that its real effectiveness lies largely in testing nearly finished Web

sites for certain performance-related issues. This is not a problem in and of itself, but such an orientation can serve a kind of metonymic function by which the whole complex of usability comes to be represented exclusively by certain instrumental aspects of it. This is a dangerous (if popular) ideological condition because it occludes the rhetorical dimensions of Web site design and usability. That is, as opposed to helping students understand that evaluating effectiveness online requires multiple approaches, SiteInspector encourages students to see their designs as altogether usable if it produces a positive technical report. As one of my students once told me, "If the computer says my Web site's usable, then it must be usable." Although SiteInspector performs certain monotonous functions rather well and can help students think carefully about some of the issues related to access, it cannot stand in for more rhetorically focused usability practices.

The site Debbie tested with SiteInspector is the Web site for the National Council of Teachers of English (www.ncte.org). There are two reasons why she selected this particular site. First, Debbie is interested in pursuing graduate studies in computer documentation and most of the programs in which to do that are technical communication programs housed in departments of English. So she used this assignment to learn more about the work culture she plans to join. Second, for the final project in my course I ask students to focus on an artifact that goes beyond the traditional parameters of computer documentation (e.g., tutorials, help systems). Terry Winograd (1999) argues that "There is no boundary at which the interface stops and the documentation begins. In a way, the history of user interfaces is one of moving more and more of the documentation responsibility onto the interaction designer" (5). If Winograd is right in this regard, and I believe that he is, then I want to prepare my students more broadly as interaction designers who can think about user advocacy issues in the larger context of an interface that a documentation system supports. To achieve this instructional objective, students must learn to see documentation as the entire symbolic complex that contributes to the usability of an artifact. This complex includes documentation, but also, for example, the design features associated with issues of access and the representations of users and their tasks. Taking procedural materials out of the picture in the final assignment helps students imagine a larger purview for documentation specialists.

*Figure 2: Summary Report Generated by SiteInspector*

Debbie tested the usability of NCTE's Web site with SiteInspector, and its reports illuminate the strengths and weaknesses of this online utility. Four of the seven reports provided usability data that would be extremely tedious to collect manually, so Debbie notes that SiteInspector clearly saves time and energy during the development process, especially for designers working on large-scale projects. For example, in checking the validity of the HTML source code underlying NCTE's Web site, SiteInspector identified 31 errors and 12 warnings, for a total of 43 lines of code requiring attention. But SiteInspector does not just detect and locate coding problems, as Debbie points out; it also explains how to fix them, therefore serving as a relatively powerful debugging tool. Thus, each entry in the report on HTML validity lists a source code line number (e.g., 448), the invalid HTML on that line (e.g., </b> </font>), and a description of the error (e.g., unmatched </B> no matching <B> seen). In this case, there is a logical problem which is straightforward: an opening HTML tag is missing (the bold tag requires both opening and closing syntax).

SiteInspector also saves time and energy by estimating site load times for five different connection speeds: 14.4K, 28.8K, 56K, 128K, and 1.44MB. Debbie realizes that such data is extremely useful because Web sites should accommodate users working with a wide range of technologies, not just the latest and greatest ones. At a connection speed of 56K, for example, NCTE's site loaded in approximately 15 seconds, a performance time that could be improved by optimizing seven images identified by SiteInspector. Conducting such an optimization test manually, however, would require designing a usability protocol and running the same computer on three different modems, an Integrated Services Digital Network (ISDN) setup, and a T1 digital transmission line. Needless to say, Debbie realizes that this test cannot be run easily in most academic courses.

Debbie interprets two more reports that indicate another strength of SiteInspector: it can help improve Web site access beyond the issue of load time. The report on Submit It! Readiness (Submit It! is a commercial service for announcing and indexing Web sites) provides suggestions for constructing and refining Meta tags, to increase the likelihood that a site will be catalogued effectively by well-known search engines. Debbie knows that there are numerous Meta tags available in HTML, two of which are particularly important. As the name implies, the Description tag allows designers to control how search engines describe their sites. For search engines supporting cross-references, the Keywords tag lets designers define alternative terms users might be employing to find a site, including typical misspellings. Debbie also knows that crucial to indexing a site is the Title tag. Although not a Meta tag, search results display the contents of this tag when listing a page. NCTE's Web site scored well in Submit It! Readiness, garnering a perfect usability mark, but Debbie can still learn about the orientation of SiteInspector

by studying the report.

The report displays the three tag types, so designers can review their contents, and it enumerates the number of words and characters in each one. As SiteInspector advises, ideally a Title tag should contain between five words and 200 characters and Description and Keywords Meta tags should contain between five words and 1,000 characters; all of this text should be as descriptive as possible. Designers wanting to edit their Meta tags can use an integrated program that automates this process, which is useful for generating a rough revision. In addition, however, the report notes if Meta tags are missing or if pages are untitled, two common mistakes in coding HTML that Debbie is alert to because they affect site access in central ways. In fact, Debbie is aware of the research estimating that only a quarter of the pages on the Web actually use Description and Keywords Meta tags. She also knows that if Meta tags are missing, search engine robots use the first 200 characters of a site to generate a crude description of it that often fails to identify accurately the purpose and content of the site.

The other report related to access specifies page elements appearing differently across platforms (Netscape Communicator versus Internet Explorer). Compatibility issues are thorny ones for documentation designers, and Debbie is very clear on this point. Indeed, Web sites not only display differently across platforms, but also across various versions of the same browser, a concern SiteInspector does not address. In this usability area, Debbie realizes that a tension exists between taking advantage of the advanced features of the newest browser and making a site browser compatible, so that it supports an acceptable number of users. NCTE's Web site rated poorly in browser compatibility, although Debbie figures out that the problem is not as severe as the report would lead her to believe. The report is organized into Internet Explorer incompatibilities—there were none—and Netscape Navigator incompatibilities, which numbered 57. Like the Submit It! Readiness report, it provides the source code line number containing the error and a description of it. But what is deceiving about the large number of compatibility problems identified is that they were caused by just a few coding patterns that could be altered easily. Although the report conveniently lists HTML tags and attributes not recognized by Microsoft and Netscape browsers, Debbie notes that a more helpful report would aid analysis further by categorizing the errors according to type.

The advantages of SiteInspector are evident to Debbie, and she would encourage her peers to use this Web-based utility regularly. But she would also stress its weaknesses, for on some level the usability perspectives informing SiteInspector are limited in ways that students of documentation should find problematic. The (last) report on popularity illuminates one shortcoming of SiteInspector, although this is not the aspect that Debbie is truly worried about. In estimating the extent of an online presence, SiteInspector relies on AltaVista, a popular search engine, to determine the number of links pointing to a Web site. For NCTE's site, AltaVista reported zero links, an ambiguous result because the Submit It! Readiness report told Debbie that the site uses Meta tags effectively. Thus, the popularity report, in a broader sense, calls attention to the fact that search engines notoriously cover only a decreasing fraction of the pages on the Web. Thus, Debbie suggests that this test could be improved by using a meta-level search environment that exploits the power of multiple engines, such as Metacrawler or Sherlock.

But the really troubling side of SiteInspector, as Debbie points out, is its usability orientation. The reports are doubtlessly beneficial, but she notes that the diagnostic tests only carry out an array of summative evaluations. That is, the technical performance data provided by SiteInspector is useful for Web sites in a near-finished stage of development: checking for spelling mistakes, broken links, and indexing concerns are tasks that should not interfere with planning and prototyping activities. In contrast, Debbie knows that rhetorical perspectives on usability extend the late-stage approach of SiteInspector, arguing that designers should always intersperse their planning and prototyping activities with formative evaluations that contribute directly to the construction of a site. Debbie understands that viewing usability testing as an integral part of site development has obvious advantages: it enlists actual users or their surrogates early on in the design process, it helps determine adjustments that might be needed prior to significant work investments, and it discourages closure on interface designs until they have been clarified and refined in iterative ways.

If the evaluations performed by SiteInspector are singularly summative, Debbie also notes that they tilt toward appraising technical performance, a system-centered bias that disregards a multitude of pivotal rhetorical issues. SiteInspector checks the accuracy of HTML code, but what does it report about the rendered

content? Is it valuable? Accurate? Compelling? SiteInspector determines how quickly users can load a page at different connection speeds, but what does it illuminate about the problems they are trying to solve? About their purposes for loading a site? About site goals? SiteInspector unearths elements that display differently across browsers, but what does it observe about the organization of site elements? Are they unified? Are they arranged in user-oriented fashions? Are they linked by discernible navigation structures? Debbie emphasizes that such rhetorical considerations are utterly unaccounted for in SiteInspector.

Proponents might argue that the usability orientation informing SiteInspector is valid, that no one utility can test for all design considerations, and that at any rate the kinds of evaluative judgments computers can promote are necessarily limited. But such an argument, in spite of its merits or flaws, is nonexistent, entrusting users to decipher the ways in which SiteInspector might be genuinely advantageous. Yet Debbie argues that the creators of SiteInspector could have been more explicit about its focus. Mapping the diagnostic tests along a usability continuum would illustrate the development points at which SiteInspector could be helpful, and evaluation narratives could articulate particular strengths and weaknesses. For example, Debbie explains that the popularity test is useful once Meta tags have been created and a Web site has been registered. However, it often takes search engines in excess of six months to index new sites, so the popularity test should be run routinely. Moreover, she notes that the evaluation narrative could also stress that designers wanting to learn more about their online presence should study server log files, which record site usage patterns, as well as conduct contextual interviews with real users. Such a situated approach, Debbie reasons, would not only educate documentation designers but also benefit SiteInspector, whose creators would be advocating a more rhetorically oriented vision of Web site design and usability.

To summarize what Debbie has learned about computer documentation, her knowledge basically falls into three categories: technical, conventional, and rhetorical. An example of her technical knowledge is that she understands the incompatibility issues associated with designing for different versions of Web browsers that run on different computer platforms. An example of her conventional knowledge is that she understands that Web sites must be designed in ways that allow them to be accessed easily by both search engines, which require HTML conventions, and computer users, who rely on discourse conventions. And an example of her rhetorical knowledge is that she understands the limitations of formative and system-centered usability methods. I think most of us would be proud to have a student like Debbie represent our programs.

But Debbie has a significant social blind spot, one produced by an education in which much of the instructional agenda seems to be little more than indoctrination into the value systems of the dominant computer culture. Put another way, her sense of user advocacy does not include an attention to the politics of technology. What do I mean by politics? I basically mean, in the words of Langdon Winner (1986), the "arrangements of power and authority in human associations as well as the activities that take place within those arrangements" (22). Given this definition, one objective of a socially informed pedagogy for computer documentation would be to draw attention to the non-neutral dimensions of computers and their non-neutral contexts of production and use. In other words, the educational system would introduce documentation students to the notion that computers necessarily instantiate values and align with preferred sets of perspectives. This is one of the directions Johnson-Eilola calls for in his argument to make technology less transparent to users.

The good news is that courses are available with such a critical focus. In departments of English and across the humanities curriculum, one can easily find courses that critically assess the various intersections between technology and society. Although these courses are hardly novel, an emphasis on technical, conventional, and rhetorical issues has tended to define social aspects not related to functionalism as outside the scope of computer documentation instruction. An example of the type of critical course to which I refer was described by Lee Brasseur in her 1993 article, "Contesting the Objectivist Paradigm: Gender Issues in the Technical and Professional Communication Curriculum." This course, which was designed for students at Illinois State University, critiques traditional notions of scientific rationality and objectivity and their gendered epistemological assumptions. More specifically, the course considers the implications of traditional discourse models from the perspective of various strands of feminist theory. Students are therefore introduced to the limitations in the Shannon and Weaver mathemati-

cal model of communication that Johnson-Eilola is so worried about. In fact, they are introduced to the gendered worldviews the model relies on, which, as Brasseur notes, "promote a disregard for the human subject that is problematic for any culture or cultures outside of the dominant one" (114). As you might have thought, this is not the kind of social critique my exemplar student Debbie has been educated to produce.

But if Debbie had taken this type of course, she would be able to identify the ways in which modes of representation can, for example, reflect and reinforce gender stereotypes. Let me elaborate with a much briefer illustration that should model the reasoning process of someone who has successfully completed a course like the one Brasseur describes. Girlhoo.com (Figure 3) is a Yahoo-like system that Debbie would interpret as detrimental to women. In structure and style it unabashedly imitates Yahoo, perhaps the most popular of all the search services, but this apparent infringement of intellectual property is not an instance of social stereotyping (I say apparent because it could be argued, weakly in my opinion, that Girlhoo is a critical commentary on the male-oriented structure of Yahoo). Rather, the concern Debbie recognizes is that the organized content in this thematic search engine tends to shore up essentialist notions of women. The feminist scholars Brasseur covers in her course (e.g., Donna Haraway, Dale Spender, Sandra Harding, Evelyn Fox Keller, Mary Lay) warn students about representations that totalize women as homogenous, unconflicted, or unified audiences. But the white, middle-class orientation of Girlhoo does just that, and the links that sex-type social and intellectual activities cast women into largely traditional roles. Users of Girlhoo happen upon no shortage of pointers to sites about beauty, fashion, Prozac, relationships, breast implants, and the like. One of the more comprehensive site areas focuses on the kitchen. It is true that more serious search engines can be found. For example, the search directory WWWomen.com covers different strands of feminism, careers in science, medical research, activist groups, and influential women in politics and history. However, as Debbie argues, regressive sites like Girlhoo, which are abundant, reflect and reinforce gendered perspectives that are harmful to women. Such perspectives, it should be noted, can be found without much difficulty in the context of documentation (see Bernhardt, 1992; Sauer, 1994).



**Figure 3: Girlhoo Search Engine**

The social blind spot encouraged by conventional courses in computer documentation is not that unusual among students at both the undergraduate and graduate level. Yet I contend that it is a significant barrier to the future vision that Johnson-Eilola proposes. To reiterate, Johnson-Eilola wants us to instantiate collaborative models of support that leverage the expertise of users, help users read interfaces as interested maps that shape both work and the worker, collapse the binary division between procedural and conceptual knowledge, and deal with complexity as an attribute and not a flaw of communication. But how can we get there from here? One key is to re-imagine our curricular designs in a way that is more expansive, to include, for example, the type of critical course taught by Lee Brasseur at Illinois State. For we cannot expect the next generation of documentation specialists to enact a social vision without an educational background that is socially informed.

So let me conclude with an example of such an educational background, one that is attentive to social as well as technical, conventional, and rhetorical issues in computer documentation. I should preface this discussion by saying that there is no one perfect approach: all programs should draw on their institutional strengths and the strengths of engaged faculty members. At the same time, local conditions cannot be the only consideration, because a program that over-emphasizes either

practice or theory will not prepare students to be agents of positive change. My curricular example is the requirements for the technical writing minor at my home institution, Penn State. I currently direct this minor, whose curricular design has been influenced by at least two things. First, the English department at Penn State has a strong tradition in rhetorical studies, which accounts for the fact that various rhetoric courses count toward the minor, even when those courses have no direct link to issues in documentation. Second, this loosely articulated curriculum is justified by qualitative research on the ways in which writers tend to develop on the job (MacKinnon). Our belief is that workplace writers become increasingly effective as they learn more and more about the contexts in which their discourse is produced and used. Because we cannot recreate the richness of these contexts in any meaningful or sustained fashion, we tend to focus on broader frameworks, although like Boiarsky and Dobberstein we use academic settings to anchor our pedagogical activities. In this way, we do not attempt to simulate the workplace and its complexities and then have students write documentation for this simulation. Instead, we attempt to provide students with a wide range of conceptual tools that can help them plan, design, and evaluate documentation systems for any specific situation.

The technical writing minor at Penn State is comprised of 18 credits. All students must take English 418, Advanced Technical Writing, which I teach as a conventional course in computer documentation. The additional 15 credits are spread out across the following three categories: 3-6 credits related to rhetorical studies broadly defined; 6-9 credits in writing and editing; and 3-6 credits related to instructional writing broadly defined. If I was advising my exemplar student Debbie with Johnson-Eilola's suggestions in mind, I would recommend a curriculum very much like this one:

- **English 418, Advanced Technical Writing.** Prepares students to plan, design, and evaluate computer documentation.
- **Speech Communication 440, Systems and Theories of Human Communication.** Analyzes biological, cognitive, social, and cybernetic theories in interpersonal, organizational, and mass communication systems.
- **Science, Technology, and Society 408, Cultural Foundations of Communication.** Examines oral, scribal, print, industrial, and electronic cultures;

analyzes the impact of technology on communications and social structures.
- **Psychology 421, Advanced Cognitive Psychology.** Explores complex mental processes: thinking, problem solving, imagery, symbolic behavior, information processing, attention, artificial intelligence, and language.
- **Art 270, Beginning Graphic Design.** Introduces students to the practice of graphic design.
- **English 495, Internship.** Internship in computer documentation

Figure 4 maps these courses onto the pedagogical landscape of technical, conventional, rhetorical, and social concerns.

|  | Technical | Conventional | Rhetorical | Social |
|---|---|---|---|---|
| ENG 418 | • | • | • |  |
| SPC 440 |  |  | • | • |
| STS 408 |  |  | • | • |
| PSY 421 |  | • |  | • |
| ART 270 | • | • | • |  |
| ENG 495 | • | • | • | • |

*Figure 4: A Balanced Curriculum for Computer Documentation Students*

This six-course curriculum models an approach that prepares students to think in expanded ways about computer documentation and its contexts. The courses cover theories of communication, the politics of technology, the knowledge types that constitute human cognition, writing, graphic design, and more. The challenge, of course, is to pull these areas together for students of computer documentation. One way to do that is to require a report that reflects critically on the internship experience (see Johnson). Johnson-Eilola reminds us that what seems obvious and natural in computer documentation is not necessarily so, that our practices and artifacts reflect certain ways of knowing and working. So there is no absolute reason why documentation must isolate users from each other, conceal its politics, separate out functional and conceptual instruction, or misrepresent the complexity of communication and work. But in order for things to change, the next generation of documentation specialists must be prepared to challenge our common-sense under-

standings of documentation and its contexts. One way to encourage that is through a socially informed pedagogy that both compliments and complicates technical, conventional, and rhetorical approaches.

## References

Bernhardt, S.A. (1992). The design of sexism: The case of an army maintenance manual. *IEEE Transactions on Professional Communication*, 35(4), 217-221.

Boiarsky, C., and Dobberstein, M. (1998). Teaching documentation writing: what else students—and instructors—should know. *Technical Communication*, 45(1), 38-46.

Brasseur, L.E. (1993). Contesting the objectivist paradigm: gender issues in the technical and professional communication classroom. *IEEE Transactions on Professional Communication*, 36(3), 114-123.

Johnson, R.R. (1996). Tales from the crossing: profes-sional communication internships in the electronic workplace. In P. Sullivan and J. Dautermann (Eds.), *Electronic literacies in the workplace: Technologies of writing* (238-252). Urbana, IL: National Council of Teachers of English.

MacKinnon, J. (1993). Becoming a rhetor: developing writing ability in a mature, writing-intensive organization. In R. Spilka (Ed.), *Writing in the workplace: New research perspectives* (pp. 41-55). Carbondale, IL: Southern Illinois University Press.

Sauer, B.A. (1994). Sexual dynamics of the profession: Articulating the ecriture masculine of science and technology. *Technical Communication Quarterly*, 3(3), 309-323.

Winner, L. (1986). The whale and the reactor: A search for limits in an age of high technology. Chicago: University of Chicago Press.

Winograd, T. (1999). Documentation, interaction, and conversation. *The Journal of Computer Documentation*, 23(4), 3-7.