

Twidale, M.B. and Ruhleder, K. (2004). *Over-the-Shoulder Learning in a Distance Education Environment*. In C. Haythornthwaite & M.M. Kazmer (Eds.) *Learning, Culture and Community in Online Education: Research and Practice*. NY: Peter Lang. 177-194.

Over-the-Shoulder Learning in a Distance Education Environment

Michael B. Twidale & Karen Ruhleder

One of the greatest challenges inherent in any distance education environment is the development of a support infrastructure that offers sustained, high quality support to all members of the distributed community. LEEP (Library Education Experimental Program) students receive extensive support from staff and other students via e-mail, Web boards, and phone conversations. Students and alumni cite this support as something that sets the program apart from others, making it valuable not only for the technical help offered but also for moral support so students do not feel that they are out there alone with a problem (Kazmer, 2000). The extensive personalized support is crucial. Although students use a standard set of LEEP technologies and are required to conform to some hardware and software standards, the problem of technical support is complicated by all the factors that neither students nor technology people can access, much less control. Students doing coursework from home often share their computers with others and rely on Internet service providers of variable quality; others do all or some of their class work at a workplace where they depend on internal networks and gateways to reach the Internet.

We would like to use this chapter to highlight some of the technological challenges. The LEEP program has met many of these challenges by tailoring existing technologies or developing in-house software to make sure that the technical infrastructure supports the goals of the underlying pedagogy (Gengler, this volume). The LEEP staff offers an

extensive set of workshops during the initial on-campus orientation to make sure all students have a common base of understanding of the applications they will be using. Faculty are given a great deal of support in redesigning courses for an online environment (Smith, this volume), and the LEEP staff are there throughout the semester to handle technical problems, especially during live lecture sessions.

Yet programs such as LEEP are challenged beyond traditional expectations by the need to provide high-quality technical support under the following conditions:

- *Distributed instructors as well as students.* The distributed nature of the program means that practitioners with special expertise can be invited to teach in the program. Technical support must thus encompass their needs as well from course development to delivery.
- *Diverse technical settings and support.* Because of the diversity of settings in which students participate in class, they may have other forms of technical support available to them. They may be able to draw on local coworkers or employers' technical staff for help with problems. Technical support at home may come from a partner or children.¹
- *Multiapplication environment:* LEEP is a multiapplication environment in which support must be able to address problems at that cross applications. Incorporating images into documents, creating a group-accessible space on someone's server, streaming video for a group presentation—all are legitimate activities that can require assistance.

The complexity of this environment places great demands on the support infrastructure. This is true not only for LEEP, but for any distributed work or learning

setting with these characteristics. We believe that the whole range of problem solving, from personal attempts, to asking colleagues, to drawing on technical support staff, needs to be understood better in order to facilitate the problem-solving process. In the work we report in this chapter, we observed how people helped each other solve computing problems in face-to-face settings in order to see what lessons we might apply to help-giving at a distance.

Over-the-Shoulder Learning

How do people solve technical problems as they arise? Online help functions, tutorials, and so forth, imply that help-giving or problem solving is a straightforward exercise in knowledge transfer: One person encountering a problem looks up the answer in the online help index; another sends e-mail to the help desk and the return e-mail solves the problem. But is the process always so clear-cut? Consider the following example, which is based on a real episode we observed (as are other examples below):²

Arthur's³ office just got a new printer that can print double-sided pages. He wants to print out a memo using this new feature, but he does not know how to do it. Bertha walks by and he asks if she knows. She has never done it either, but overheard someone else saying that there was 'a box you can click so it will do two-sided.'

In the process of figuring it out together, Arthur mentions that he hopes he will remember this when he prints the monthly report. Bertha looks at him and says, 'Did you know that there's a new format they want us to use for that?' Cristobal overhears the comment and says that he just got a memo with instructions for the new format.

Cristobal hands Bertha the memo. She skims it and says, 'So we have to do different margins for it, too?' Cristobal answers, 'Yes, for odd and even pages so they can put it in

a binder.’ Arthur says, ‘I think I just saw something like that while we were looking for the box for the printing.’

We term this kind of collaborative problem solving *Over-the-Shoulder Learning* (OTSL). This kind of problem solving takes place in response to a computer problem that arises while trying to get work done. Help can come from a designated technical support person, from a technically adept coworker, or even coworkers who do not perceive themselves as “techies.” Working and learning are merged, and knowledge about the task and the broader context in which it is being carried out is as important as technical knowledge.

This certainly describes the case above. Arthur had a real memo he wanted to print double-sided. He and a coworker sat down to pool their information about this feature and printing in general in order to figure out how to do it. A call to technical support might have done just as well. The importance of context, however, comes in the second paragraph when Arthur mentions the monthly report. The technical problem was really a work problem in disguise. And although Bertha’s comment about the new format and Cristobal’s provision of the instruction memo may spawn other technical questions, they will arise in a work context in which getting the report out is the primary goal.

OTSL can also propagate knowledge within an organization. In fact, LEEP offers us an example of cyber-OTSL. One now famous feature of the chat room is that students can *whisper* to each other. On the right side of the screen, a list of names shows who is logged in to the chat room at that time. Anyone logged in to the chat room can click on an individual name, highlighting it, and send a message—a *whisper*—that will be seen only by the selected person. This feature was discovered by one student who had used it

in another chat application. He tried it, it worked, and knowledge of the feature disseminated throughout the LEEP population.

Help-giving and peer support for information technology problems has already been studied extensively (see Twidale, 1999, for an overview). However, most studies have concentrated on whom the help-seeker chooses to ask for help, why that person is chosen, and the details of their physical, social, and organizational proximity. In our work we focus on the interactions themselves, such as the help-giving episode portrayed above. Our goal is to draw conclusions about the nature of OTSL and develop interfaces that will facilitate collaborative learning (Eales & Welsh, 1995). The next section provides more detail about the study we carried out.

OTSL in Context

We undertook a series of field studies in a variety of different contexts in order to develop a data set of help-giving interactions. The primary sites, where we made multiple observations over many months, included:⁴

- A university library reference desk
- A university library help line for recalls, billing questions, etc.
- A corporate research lab in which interns worked on engineering design projects
- A technical support team of large academic outreach department
- Three groups at the headquarters of a large financial services company: one doing *in-house systems development*, one responsible for *technical training*, and one providing *long-distance phone support* for branch managers

While our studies examined informal learning in workplaces, there are many parallels between these sites and LEEP. The activities at the financial services company, for

example, sound very much like the work the LEEP staff does, particularly the real-time response that groups like the one providing long-distance phone support must provide. Interns in the corporate research lab ran into problems that involved settings and configurations at the application or system level. The technical support team dealt with problems involving servers and networks over which they had no control. At all of these sites, people had to cope with multiple applications, short- and long-term deadlines, and incomplete understanding of the system as a whole due to its complexity and the dynamic nature of computing.

There are also many similarities between LEEP and the workplaces we studied in the nature of the people we observed. Most participants had many years of work experience and were often are highly proficient in using computing to carry out a set of work tasks and routines. This was true of their coworkers and supervisors as well. Many had access to training through their workplace and were motivated to share tips and talk about problems. In short, they had all the makings of willing learners and problem solvers. Similarly, most LEEP students are also holding full-time jobs in contemporary workplaces where computing is a requirement.

Our primary method of data collection was observational. Corporate sites, particularly those engaged in research and development (R&D), were generally willing to let us observe but not willing to allow us to do video- or audiotaping. With the help of graduate and undergraduate assistants,⁵ one or two people would spend several hours at a time at a particular site taking notes on any help-giving events they observed and talking briefly with the participants afterward if possible. Outside the corporate settings, we were able to audiotape technology help interactions by users in an academic department over a

period of about six weeks as they were happening. In addition, we opportunistically studied ourselves and our colleagues engaging in OTSL, with some episodes captured on videotape.

OTSL occurred in all the contexts we observed. In fact, despite limitations and biases, disparities in terms of organizational size, users' level of technical training, sophistication of available computing equipment, the short- or long-term nature of the work, or any other set of factors, the *commonalities* were most striking. We discuss these commonalities, and their implications for technical support in distance environments, in the following sections.

Locating the Problem

Although computer problems may already be brewing under the surface, they become apparent when something looks wrong (e.g., a printout, a screen layout) or when an application or device responds in an unexpected way. Sometimes the fix will be simple. To return to the example of Arthur and his coworkers, he might have forgotten to click the correct box for double-sided or to unclick the box for a document he wants printed single-sided. But what if the problem goes beyond your understanding of the system or the application? How do you locate the problem when you see something is wrong but the potential cause is hidden from you?

We observed two common categories of problems at all of our field sites. In the first category, the problem arose when something didn't look right to the user (e.g., a document, or a three-dimensional rendering using a CAD system), and the source of the problem lay somewhere in what we call the *substrate* (i.e., the many options, configurations, preferences, and exceptions available under various menu headings). In

the second category, problems arose when the system or network didn't act according to the user's expectations (e.g., a Web page was suddenly inaccessible, or an e-mail disappeared mysteriously), and the source of the problem lay not at the level of the application but in how that application fit within the broader architecture of systems and networks within the organization. We call this level the *superstructure*. Two examples of superstructure and substrate follow.

The Invisible Superstructure: "Where am I?"

Users encounter two kinds of location problems that often overlap. First, they often have little sense of how different applications fit together in the work process; and second, they can be unsure about where they are within a broader computing infrastructure. For example, users accustomed to thinking about computing in terms of the applications they use on their desktop computer may have trouble sorting out what is on their computer, what is on the server or the network, and how these interact with each other. For example:

Dolly is a professor with strong computer skills who has promised to show her less-skilled colleague, Edouardo, how to 'put a document on the Web.' Edouardo uses an HTML editor to take an existing document on his own computer and turn it into an HTML: file. Then Dolly guides him through the process of transferring that file from his computer to his account on the local server so that others may view it. Once Edouardo has carried out that step, he and Dolly look at the file using a standard Web browser.

Now Edouardo wants to modify something. He makes the change in the editor and saves the new version, but sees that the change hasn't appeared in the browser window. Dolly explains that every time he makes a change he must transfer the new version into his account on the server. Edouardo follows her instructions again but the change still

doesn't appear. Dolly explains that he also has to click on the Refresh button on the browser so that the latest version of his document will be displayed.

To Edouardo, the two documents he sees—one in the HTML editor and one in the browser window—look the same and appear on the same screen in front of him. To Dolly, they may look the same and appear in the same place, but her greater knowledge enables her to place them into a broader context, invisible to Edouardo, in which they reside in two places. We return to this issue later in the paper.

The Invisible Substrate: “Why is it doing that?”

Another kind of invisibility entails the many configurations, options, preferences, characteristics, and customizations that users may not even be aware of, but that shape the way applications function. For instance, there are system configurations that determine the availability of access to networks, peripheral devices, and so forth. There are printer settings that may override document settings. Applications give users the means to tailor the look of a single document or to customize it in a way that affects subsequent documents as well. An example of this follows:

Fay is writing a proposal using Microsoft Word.⁶ She adds page numbers by pulling down the Insert menu, dragging the cursor down two lines to Page Numbers... and selecting the position and alignment she wants.

Now Fay decides that she also wants to insert a header. She goes back to the Insert menu but doesn't see a menu item for adding a header. Under AutoText, however, she spots something to do with headers and footers and tries out the first option it lists: - Page -; the text “- 1 -” appears where her cursor was located, in the middle of the first page of the document.

Surprised, Fay seeks out Gustav, one of the technical staff, who goes to her office with her. He tells her to delete the text that she accidentally inserted and to go to the View menu instead. Fay clicks on View, pulls down the cursor until it highlights Header and Footer, and clicks again. Dark dotted boxes appear at the top and bottom of each page, the text of her document has dimmed, and a tool bar has appeared. Gustav explains to Fay that she can type into the boxes, but in order to return to editing her document she has to close the header/footer view from the tool bar.

Fay's confusion stems from the (very reasonable) assumption that she cannot view something she has not yet put into the document. If she has used footnotes in the past, for example, she would have gone to the Insert menu to enter a new footnote, then used the View menu to view and edit the footnote text. At any rate, Fay now has another set of steps to follow even if they don't seem logical to her.⁷

Gustav's technical training, however, has given him a different perspective on how an application might structure a document internally. For instance, a Word file has some inherent properties that the user cannot change but can ignore or modify to some extent. Gustav knew that all Word documents have headers and footers even though they are not visible unless the user edits them.

Collaborative Problem Solving

Despite the complexities outlined above, people do cope with the breakdowns that occur in their use of computers. We outline here some of the mechanisms that enable this coping. Pervading all the mechanisms of collaborative technical problem solving is the importance of understanding the local context.

Active Participants

We have observed that people asking for help have generally attempted to solve the problem themselves first. They then try to explain their earlier attempts at the outset of any help-giving interaction. This explanation seems to be in part to justify why they deserve to ask for help, and also to give the help-giver some clues about the nature of the problem in order to speed up the process. Even in the simplest case where the help-giver knows the solution, help seekers are usually very active participants in the conversation, trying to clarify their own understanding. However, in cases where the solution is not immediately apparent, it is particularly noticeable how collaborative the problem solving is, with the help-seeker volunteering information on things that they have tried, giving more background on the overall goal, and/or discussing the suitability of alternative work-around solutions. This two-way process of discussion, shared computer operation, and gesturing at the computer, is a way of trying to make visible those invisible layers described above. Viewing the interaction as provision of a simple fix by a help-giver, or as knowledge transfer, misses the point: help-giving episodes need to be understood as collaborative, interactive processes where both sides learn something.

Acknowledging active participants carries implications for the underlying ethos of the community as one that encourages asking for and giving help and an open exploration of technologies, and indeed encourages openness about one's own confusion (Gengler, this volume). For example, if students are to be encouraged to try and solve problems themselves before asking for help, we must be willing to deal with those attempts appropriately. Sometimes their attempts will worsen the problem. If students feel they are being blamed for their prior attempts in the problem-solving interaction, next time they

encounter a problem they may not try themselves, resorting first to technical support or even just giving up. Given the interactive nature of technical problem solving, interpersonal skills can be at least as important as technical skills in determining successful efficient interactions that contribute to overall effectiveness (a solution that demoralizes the help-seeker may be effective in the short term but have severe long-term consequences).

Understanding the Real Goals

While a person may ask for help in using a particular feature of an application (e.g., a desktop publishing application), if the help-giver can determine why they actually want to use this feature (e.g., to produce a certain effect for a particular assignment), then an alternative feature may be proposed or even a way of achieving the desired result by using a combination of applications. These are ingenious *workarounds* (Gasser, 1986) that can be much more efficient than trying to solve the initial problem as described. For a successful workaround to be possible, the help-giver needs to understand the person's real goals. Only then can the two participants work together toward the best solution. This is why asking a colleague for help can be so powerful, even in cases where the help seeker also has access to a more remote but more skilled source of purely technical expertise. The conversation with the colleague can be carried out in terms of the work to be done (e.g., producing and formatting a report in the appropriate way for a particular recipient), and proceeding from there to look at different technological solutions or workarounds. Thus, the technical problem should not be viewed just as a technical problem, but as one that also includes the larger work context.

Solutions Often Involve Multiple Applications

One finding that particularly surprised us in our observations was how often people's work tasks involved using more than one application, and how often the solutions to problems also involved more than one application. For example, one task involved copying a discussion from a live text chat discussion into a word-processing package, editing it into a summary, and then posting that text to a bulletin board.

It appears that one of the strengths of the modern personal computer is that it involves the ingenious composition of functionality from multiple applications in order to fulfill a higher-level work goal. Multiple application composition can be seen in the careful development of the LEEP technology infrastructure (Gengler, this volume). Technologies are composed to create particular learning experiences to fulfill the pedagogical goals of a professor. Similarly, students compose applications to produce innovative pieces of work to fulfill the requirements of an assignment.

This is highly desirable and to be encouraged. It creates substantial robustness in practice: if the needs of teaching or learning change in a small way (e.g., application to a new, more meaningful context for a student to apply a theory, and hence test their understanding) or in a big way (a new kind of course is offered), people can recombine their applications. Similarly, if a new useful functionality appears in a currently used or brand-new application, it can be incorporated into the overall activity by changing the way that applications are combined to achieve the same main goal.

The significance of this for help-giving is that conventional application help (online help manuals and other support infrastructures) discuss only that one application. Unfortunately, if many work tasks involve using several applications together to get the

work done, the help system is unlikely to describe how to do such multiapplication tasks. Furthermore, the tasks are likely to involve many context-specific aspects. Thus, even an integrated suite of applications (such as Microsoft Office that might include examples of multiapplication activities within the suite) is unlikely to describe how to do the desired context-specific task. It is only by drawing on local expertise that local solutions can be developed and propagated.

Technical Problem Solving in Distance Education

Distance learning presents a slightly different context for technical help giving and problem solving than face-to-face settings. We will consider the distance education technical support issue from the perspective of the GSLIS LEEP program, which we are familiar with (one of us having taught in it for the last six years, and both of us having done research on the program).

Many aspects of LEEP are typical of the workplaces we studied. The use of the various applications is a means to an end. Students must learn about getting online, using e-mail, chat, audio, bulletin boards, Web pages, and databases, in order to take classes and obtain their professional degree as a way of furthering their career. As well as studying and participating synchronously and asynchronously in class, students must learn how to use various bibliographic databases, specialist applications, and more standard applications in order to produce documents such as reports, term papers, and Web pages. All this implies acquiring a substantial amount of technological skill, and by some people who, although intrepid enough to be pioneers in distance education, would not describe themselves as particularly technology oriented.

Remote technical help and problem solving entails all the issues outlined above, along with certain additional complexities due to the geographic distribution of the participants.

Establishing and Maintaining Context

Technical problem solving between staff and user at a distance requires extra effort. First, help-givers must determine what the remote user is seeing on the computer screen. They must also determine the wider context and infrastructure layers that may have an impact on understanding the underlying problem and routes to a resolution. These can range from understanding the underlying work goal to knowing about the particular hardware setup, operating system, modem used, ISP, local peculiarities, and so forth. Furthermore, context can change over time. A student may be using their home computer, or their work computer behind a firewall, and need to switch easily between the two.

Impoverished Resources for Supporting Conversation

Remote help giving is usually accomplished via telephone or text chat if synchronous, or e-mail or bulletin board if asynchronous. All these modes of interaction are verbal and/or textual in nature, which may be fine for certain kinds of discussions, but at the least is a limitation compared to the multiple cues available in a face-to-face interaction. In the latter we see much pointing and gesturing at the screen with fingers, pens, and the mouse cursor. For example, talking a person through the operation of a graphical user interface is much easier if one can point to the icons and active areas on the screen as they are discussed. Discussion becomes much more fiddly if one has to use words to refer to graphical elements, particularly in cases where the help seeker does not know the name

or meaning of that element. In addition, in face-to-face interaction nearby artifacts may be used to support discussion, such as paper printouts, manuals, work documents, or sketches on paper. All these can be used to help to illustrate what is wanted or has already been tried. Problem-solving conversations are about trying to make visible relevant aspects of the invisible substrate and superstructure. The computer screen and other resources can all help in that process.

Barriers to Turn-Taking in Computer Operation

As well as difficulties in seeing the same screen and what is pointed to on it, it is either difficult or frequently impossible to easily switch control of the computer back and forth between participants, an activity that we see in face-to-face help giving, particularly during mutual problem solving. Even conversational turn-taking, so much a feature of face-to-face help giving, is somewhat more clumsy over an audio channel in the absence of nonverbal clues of expression, gesture, and pointing.

The LEEP Approach to Technical Problem Solving

Given all these difficulties, remote problem-solving interactions are likely to take longer and so are more costly than face-to-face ones. Therefore there may be a temptation to minimize the amount of technical help provided, and instead require students to draw on their own resources. And yet it seems that LEEP students are very happy with the amount and kind of help they receive from the Instructional Technology Office (ITO) and from their peers. How is this achieved? We see that a complex supportive ecosystem has evolved. We note some of the aspects below as they relate to our findings in other domains. Clearly this is just one part of the contributing success factors (see Gengler, this

volume, for more details on ITO's user-centered focus as a critical component). A major managerial factor is the valuing of and resources allocated to the ITO. It is the way that all the elements interrelate with each other, and also interrelate with other aspects of the LEEP experience as described throughout this book, that enable the great challenge of technical problem solving to be addressed successfully.

We know that students hold the ITO in very high regard (Kazmer, 2000). When talking about their experiences with ITO, LEEP students mention not merely the technical expertise of ITO, but their empathy and reassurance. This would seem to be important to students, and is understandable given the combination of isolation and frustration that can arise when a distance learner encounters a technical problem. As well as this very careful demeanor within their technical help giving, the ITO has a deep understanding of the higher-level goals of the LEEP students. This is because most ITO staff members are themselves GSLIS, library and information science students or graduates. As a result, if a problem is encountered with creating one aspect of a Web page for an assignment for a particular class, it becomes easier to discuss the acceptability of perhaps a completely different way of achieving the desired effect that still fulfills the requirements of the assignment. Although we have noted the value of physical proximity of colleagues in the workplace help-giving studies, it seems that the ITO manages to maintain context and a form of social proximity via a shared understanding of the LEEP program and considerable effort in community building.

An important part of the success of LEEP overall is its creation and nurturing of a supportive online community, which in turn becomes a learning community (Hearne & Nielsen, this volume). This mutual support between students involves not just emotional

support, practical advice in juggling competing demands (Kazmer & Haythornthwaite, 2001), and strategic advice on studying and how to become a distance education student, but also technical help giving and problem solving.

The whole domain of library and information science is imbued with an ethos of service, and so help giving is validated and supported. In many ways help giving is very similar to the traditional library reference interview (Bopp & Smith, 2001). The skills of good reference work have been analyzed and are taught in LIS schools, and there is a growing interest in remote reference (Sloan, 1998). An important reference skill is to understand the real information need of the patron, and how that may be different from the initial articulation of the request. Another skill is satisficing—agreeing on what kind of result will be good enough for the needs of the patron, given the available time and resources of both patron and librarian. This closely parallels the help-giving interactions we have observed. Consequently help giving becomes a part of normal activity in the LEEP learning environment, and the specialist help giving and problem solving of the ITO becomes more than an infrastructural element, no longer separate from the main content of teaching and learning.

Supporting Technical Problem Solving in Distance Education

We end with a section exploring the implications of our analysis of various workplaces and of LEEP, and the recurrent themes that emerged. For simplicity we group our suggestions into technological, managerial, and pedagogical solutions. However, as in our analysis of LEEP, we emphasize that to be effective, all three have to interact with each other and the overall sociotechnical infrastructure. Otherwise the whole point of the importance of context is lost.

Technological Solutions

Technological solutions used for the problem-solving episodes can vary in technological sophistication and hence in cost and current practicality. Some technologies are immediately useable and indeed have been used in LEEP. For example, the ITO uses Virtual Network Computing as a means to provide remote technical support. This software allows remote access to view and control another machine. Also, in teaching online searching using Dialog (by exploiting a simple Unix script), students are able to see the instructor doing a telnet-based search—a kind of virtual over-the-shoulder view (see leep.lis.uiuc.edu/support.html for details and examples of excellent online guides). Other relatively simple options to support remote problem solving include the taking of screenshots and e-mailing them back and forth as a way of establishing context, and similarly exchanging a quick sketch created in a drawing package. To be effective, such techniques need to be simple and fast enough to use that they do not get in the way of the problem solving they are intended to support.

The field of Computer Supported Cooperative Work (CSCW) involves the development of technologies to support interaction between people at a distance (see Twidale & Nichols, 1999, for an overview). Remote help giving certainly fits within this remit, and a variety of technologies have been developed and are available. These include, video links to support nonverbal communication, and shared screens and telepointers so that two people can cooperate around an application at the same time knowing that they are seeing the same thing. One CSCW application that can readily be downloaded and investigated for this kind of use is Microsoft's NetMeeting.

It should be noted that considerable evaluation work of CSCW applications has been done, and it has been found that they do not always succeed, particularly if they do not fit well into the wider context of the use environment (Grudin, 1989). Thus, a two-way video link may be very helpful to problem solving in theory, but in a particular context it may require too much set-up time to be worthwhile or require too much bandwidth to be usable. CSCW technologies attempt to make the interaction as close as possible to the situation where the people are actually together in the same room, or the help-giver is leaning over the shoulder of the help-seeker. To the extent that this is currently possible, it addresses the additional problems of remote technical problem solving outlined above. However, remember that in the first part of this chapter we showed how even face-to-face help giving has its problems.

We are currently investigating the development of new interfaces and functionalities to address gaps between help-givers and remote users (Twidale & Brady, 2004; see also Prince, et al., 1999). Modern graphical user interfaces are very good at supporting efficient use by people who don't encounter problems, and can be relatively easy to learn the basics, given a little help. If the user is confused and the solution is just one or a few clicks away, well-designed interfaces can make it easy to guess the right thing to do, and good help can indeed be helpful. But for more complex tasks, or when an attempt fails, there is relatively little provision for supporting diagnosis of what has gone wrong and why, and for supporting trying out solutions.

What would a specialist computer interface designed to support problem solving look like? How would it be different from current interfaces designed to support smooth use and guessing of the next step? We are looking at techniques for capturing sequences of

actions and then providing an easy-to-review visualization of what has been tried. The same record of actions can then be used in sharing a solution, both with the person asking for help and with others who might want to know, such as via an updatable FAQ (frequently asked questions) list. Our approach is analogous to existing screen capture programs such as Camtasia, but aims to allow greater tailoring and customization of explanations. We are also exploring whether a more maplike interface would help during problem solving when the help-giver and seeker are lost and looking for inspiration. Such a map would explicitly represent all available options and be updatable and annotatable based on what had been tried so far, and even what others had tried and found useful in the past. Another option is to provide ways to quickly compare the settings of two applications on different computers to see how they differ in order to address a common problem of a help giver complaining “but that works on my computer!”

Managerial Solutions

The main point to acknowledge is that students will have problems with technology (Ehrmann, 1999), and that a multithreaded solution is needed for the overall problem of technical support. Despite the best efforts of setting minimum technical competencies for incoming students, and providing excellent training and online information resources, technical problems will inevitably arise. Consequently it is important to provide resources to facilitate different kinds of help giving and problem solving. We are concerned that at times this provision may be overlooked or skimped because of limited budgets. Yet comfort with technology, and an infrastructure to facilitate problem solving, can be a significant contributor to students’ satisfaction and persistence in the learning environment. In the absence of a rich-enough problem-solving infrastructure, faculty may

worry that students will rely on them for technical help (Levy, 2003), not only adding to the burden of teaching online, but also having an impact in students' evaluations of their courses (Lackey, 2001).

As well as providing appropriate resources for technical support, it should be a managerial responsibility to help sustain an online learning community. There will never be the resources available for designated technical support people to solve every problem at the moment of need. Students ought to be encouraged (and helped) to try and solve as many problems themselves as possible, and to share their knowledge and help-giving skills with their peers. We concede that in teaching library and information science, the LEEP program has a distinct advantage here, since help giving is both part of the culture and part of the content of the domain. Nevertheless, help giving should be validated and encouraged regardless of the subject learned; it is a crucial skill in modern organizations that aspire to be learning organizations or to put the promise of knowledge management into effect. Simple kinds of encouragement might include extra validation or credit for help giving, support for shared problem solving, as well as moving away from paradigms of single student work to acceptance and encouragement of collaborative work.

Pedagogical Solutions

We do not believe it is feasible to rely solely on technical support personnel to help with all problems at all times. Each student who enrolls in a program brings different technical competencies and confidence. We doubt that it is possible, economically feasible, or even advisable to directly teach them all the technological skills that they will need. We are investigating how students may be taught how to investigate more kinds of problems

themselves so that they only need to resort to technical support for the more obscure problems.

One possibility is an updating of Carroll's minimal manual approach (Carroll, 1990) to apply to modern computer applications. In brief, this involves teaching people a minimal subset of the features of an application, sufficient to be able to do something meaningful, and then encouraging them to explore other features on their own. Just suggesting that they might like to explore and to share what they find with their peers is unlikely to be enough. We believe that the skills of how to explore can be taught, even to those who are unused to learning about applications in this way and are somewhat afraid of the process. These meta-learning skills need not be anything exotic and could include tips such as: create a really simple file and investigate the feature by changing one thing at a time; or try to replicate the problem, but with the simplest case you can find so that the problem still remains. These are skills that people comfortable with learning technology through exploration have often developed over time, but which are rarely explicitly taught or even articulated to those with less experience.

In some cases (many, we hope) this exploratory approach will enable students to solve a problem or uncover a new useful feature themselves, and they should also be encouraged to share this knowledge with their peers. This, in turn, requires managerial and technological encouragement, so that it is both valued and easy to do. However, some problems will always be too tricky for students to solve themselves, and a student will need to draw on a fellow student or on technical support. These interactions, although very valuable and capable of solving complex problems (as we have described) are still somewhat time consuming. If we are to encourage them, we need to consider

how to maximize their efficiency. One way is to teach the skills of asking for and giving help (Agre, 1994a, 1994b), and even to teach particular skills of remote help-asking and giving. Another way is to explicitly acknowledge why problem solving can be so fiddly (because of the difficulties involved in locating the problem, which we have illustrated). A discussion between the problem-solving participants about where the problem is and the different parts of the substrate and superstructure in which it might be hidden may also help in clarifying the discussion and arriving at a solution faster.

In summary, we believe that with appropriate managerial support, by teaching of the higher level skills of learning how to *learn* about technology and how to effectively discuss and resolve technology problems, coupled with suitable technologies to assist these discussions, it is possible to contribute to the rich technology problem-solving infrastructure that a distance education program needs. These suggestions complement the excellent technical support infrastructure already provided for LEEP by the ITO (Gengler, this volume), and suggest ways for other distance environments to approach and/or expand their own help-giving practices.

Conclusion

The technologies used in distance education offer great potential for innovative pedagogy and the creation of a vibrant learning community. This requires careful decisions about technology selection and ongoing revision. In this chapter we have stressed that technology problems will arise, and careful thought needs to be given to how to address them. We do not believe that all problems can be sidestepped by huge amounts of application training or off-loading responsibility to students. By studying a number of very different environments and identifying commonalities, we have shown how the

resolution of technical problems has a very important social component. Collaborative problem solving between peers and with technology experts, drawing on a thorough understanding of the context of use of the technologies concerned, allows people to cope and to come up with solutions. We have looked at the particular challenges of distance education and considered how collaborative problem solving can be supported through a rich mix of self-help, peer support, and the help of technology experts. In order to succeed, this mechanism needs to be thoroughly integrated into the wider context of establishing and nurturing an online learning community.

Acknowledgments

We wish to thank Vince Patone for his help in describing the work of the LEEP Instructional Technology Office. This research was supported by the National Science Foundation under Grant No. 0081112.

Endnotes

1. To the best of our knowledge, we have no recorded incidents in which conflicts between LEEP staff advice and advice from other sources led to greater conflicts in work or home environments.
2. The examples used in this chapter are derived from real episodes. The actual examples are too complex in terms of the interactions and the contextual explanations required to make sense of them. We have tried to preserve the character of the interactions, however. This particular type of help-giving episode in which people start to cluster and related topics arise, is common to all of the sites we observed.
3. We are indebted to the Champaign Public Library for providing us with the names of Atlantic hurricanes during the 2002 season to use in our examples in this paper.
4. We also have some data from sites that did not work out for one reason or another (e.g., not enough sustained activity, difficulty in negotiating times for observations). These include a small local newspaper, a campus newspaper, a public library cataloging department, and a three-person technology team supporting a medical clinic. We also collected data serendipitously from interactions observed in our own

work environment. We do not consider these as primary sites because of the lack of continuity of our observations, but they still inform our work.

5. Listed alphabetically: Varinia Godoy, Karen Medina, Alicia Oryhon, Vandana Singh, Dinesh Rathi.

6. Although we have tried to avoid identifying the names of specific applications, this example best makes sense in the context of the application.

7. If the reader of this chapter has access to Microsoft Word, we encourage that reader to access the help function, select Contents and Index, and enter *header* as the search term.

References

Agre, P. (1994a). The art of getting help. *The Network Observer*, 1(2). Retrieved from <http://polaris.gseis.ucla.edu/pagre/tno/february-1994.html>.

Agre, P. (1994b). How to help someone use a computer. *The Network Observer*, 1(5). Retrieved from <http://polaris.gseis.ucla.edu/pagre/tno/may-1994.html>.

Bopp, R., & Smith, L. (2001). *Reference and information services: An introduction* (3rd edition). Englewood, CO: Libraries Unlimited.

Carroll, J. M. (1990). *The Nurnberg funnel: Designing minimalist instruction for practical computer skill*. Cambridge, MA: MIT Press.

Eales, R. T. J., & Welsh, J. (1995). *Design for collaborative learnability*. Proceedings of the First International Conference on Computer Support for Collaborative Learning, Bloomington, IN (pp. 99–106). Mahwah, NJ: Lawrence Erlbaum.

Ehrmann, S. C. (1999). Asking the hard questions about technology use and education. *Change*, 31(2), 25–29.

Gasser, L. (1986). The integration of computing and routine work. *ACM Transactions on Information Systems*, 4(3), 205–225.

- Gengler, J. (this volume). User-centered support and technology in LEEP.
- Grudin, Jonathan T. (1989). Why groupware applications fail: Problems in design and evaluation. *Office Technology and People*, 4(3), 245–264.
- Hearne, B., & Nielsen, A. (this volume). Catch a cyber by the tale: Online orality and the lore of a distributed learning community.
- Kazmer, M. M. (2000). Coping in a distance environment: Sitcoms, chocolate cake, and dinner with a friend. *First Monday*, 5(9). Retrieved from http://www.firstmonday.dk/issues/issue5_9/index.html
- Kazmer, M. M., & Haythornthwaite, C. (2001). Juggling multiple social worlds: Distance students online and offline. *American Behavioral Scientist*, 45(3), 510–529.
(Reprinted in this volume).
- Lackey, J. R. (2001) Who is really responsible for online students' technical support? Presented at Southeast EDUCAUSE Annual Meeting, Orlando, FL. ID No SER0108. Retrieved November 2, 2002 from <http://www.educause.edu/asp/doclib/abstract.asp?ID=SER0108>
- Levy, S (2003). Six factors to consider when planning online distance learning programs in higher education. *Online Journal of Distance Learning Administration*, 6(1). Retrieved from <http://www.westga.edu/~distance/ojdl/spring61/levy61.htm>.
- Prince, R., Su, J., Tang, H., & Zhao, Y. (1999). *The design of an interactive online help desk in the Alexandria Digital Library*. Proceedings of the International Joint Conference on Work Activities and Collaboration, San Francisco, CA. (pp. 217–226). New York: ACM Press.

Sloan, B. (1998). Service perspectives for the digital library: Remote reference services, *Library Trends*, 47(1), 117–143.

Smith, L.C. (this volume). Faculty perspectives.

Twidale, M. B. (1999). *Over the shoulder learning: Supporting brief informal learning embedded in the work context*. Technical Report ISRN UIUCLIS—1999/2+CSCW. Retrieved November 2, 2003 from <http://www.lis.uiuc.edu/~twidale/pubs/otsl1.html>.

Twidale, M. B., & Brady, A. (2004). Wayfinding in an interface: Would a map help? GSLIS Technical Report.

Twidale, M. B., & Nichols, D. M. (1999). Computer supported cooperative work in the information search and retrieval process. *Annual Review of Information Science and Technology*, 33, 259–319. Medford: Information Today.