User Assistance in eLearning Applications

Julie McHam

April 26, 2015

IDT-520 Instructional Design Issues in eLearning and the Design Process

California State University, Fullerton

William Schools, Instructor

User Assistance in eLearning Applications

New learners do not always understand how to use an eLearning program. Help or Directions are often provided to help learners get started using the program, learn how to navigate the program, and how to save or exit the program. Help and Directions are just two components of a larger group of tools known as *user assistance.* User assistance (UA) is "all the manifestations of content aimed at assisting the user to use the application" (Self & Bleiel, 2010, p. 1). User assistance applies to all types of software applications, from large, enterprise-wide business applications to short, eLearning tutorials and drills. This paper discusses the evolution of user assistance; explains related concepts of minimalism, progressive disclosure, and layering; and presents recommendations for effective user assistance techniques in eLearning programs.

Evolution of User Assistance

Self and Bleiel (2010) explain that the history of user assistance started in 1988 with the Help system that was developed for QuickBASIC, an application that ran under the Microsoft DOS operating system. This help system introduced the concept of *context-sensitivity*, which connected relevant information from the Help system to the application using cross-references. Over time, Help evolved as operating systems and application platforms progressed. For example, WinHelp was developed for Windows 3.0, which used a graphical user interface. HTML Help was designed for Windows 98, and browser-based Help came into vogue as developers started to create applications that ran on web browsers.

Help systems were not always helpful. In fact, some were hard to use. "This type of assistance isn't always effective and many avoid it because it is frustrating to find and use the information" (Miller & Mc Candless, 1999, p. 32). Stahl and Bromme (2009) explain that getting help using a software application is a five step, complex process. First, users need to be aware

that they need help. Second, users must decide to seek help. Third, users have to identify who or what can help them. Fourth, users must ask for or request the help. Finally, users must evaluate the help they receive and apply it to what they are doing. Since this tends to be an overly-complex process, many users will not use help and instead ask someone for assistance or search the Internet for more immediate answers. "Learners spent very little time on actual help use. They ignored hints on how to solve mathematical problems and looked instead directly for answers. More abstract information such as definitions given in a glossary was virtually ignored" (Stahl & Bromme, 2009, p. 1020).

Over time, attempts were made to improve Help. Self and Bleiel (2010) explain how the progression of Help continued to evolve from standalone help systems that were developed separately from the software to useful information that is part of the application. For example, Help expanded to include clear field descriptions and hover text that identifies the purpose of tools and buttons. This Help, called *embedded user assistance,* "makes information more accessible and more relevant, and reinforces the readers' expectation that UA is part of the application" (Self & Bleiel, 2010, p. 2). Embedded user assistance provides users with immediate help, is specific to the task at hand, encourages learning, and takes no initiative on the part of the user.

Miller and McCandless (1999) discuss some of the early attempts at embedded user assistance including *intelligent agents* such as Clippy, the paper clip character provided with Microsoft Office 97. This character would pop up when it anticipated a user needed assistance and would offer suggestions. Many users found Clippy to be annoying and Microsoft ultimately dropped the character from their software. Other user assistance tools introduced were *What's*

This? Tools that popped up an explanation when the user pointed at something and *interactive help* that provided Wizards to assist with completing complex tasks.

Minimalism

A primary theory that has driven the evolution of user assistance is that of *minimalism*. Minimalism is an instructional design theory proposed by John M. Caroll in the 1980s. His research and subsequent writings were widely adopted by writers of software training documentation. Farkas and Williams (1990) describe minimalism as the way to ensure that users can begin working with a software application right away without having to stop working. Minimalist documentation should be "as brief as possible, support the accomplishment of real work, help learners recognize and recover from errors, and, when possible, permit non-sequential reading" (Farkas & Williams, 1990, p. 182). In its early days, minimalist documentation was highly criticized and misunderstood as being documents that were written using as few words as possible. However, Carroll promoted the idea of exploratory learning, where users tried things out for themselves instead of being told what to do every step of the way. He felt that documentation should support this type of learning so that users could have the opportunity to learn via trial and error. Carrol also believed that people who use software are just interested in getting their work done and they do not want to be bothered with lengthy tutorials.

In later years, Carrol's concepts of minimalism evolved into a set of principles promoted by JoAnne Hackos, a renowned Technical Communications expert. Gallon (2013) includes Hackos' minimalist principles in a presentation he gave on the cognitive aspects of user assistance. These principles include focusing on action-oriented approaches to documenting tasks, committing to understanding the user's viewpoint, recognizing that troubleshooting information is necessary, and ensuring that users can find the information they need. Gallon also

recommends embedding these principles into the user interface, in not just a minimal way, but a meaningful way. Gallon's solution is to provide embedded user assistance that gives users all the information they need, when they need it, and without extraneous information. The information should be embedded into the software user interface so that users do not have to search for what they need. Information users do not need should be hidden from their view.

Progressive Disclosure

Gallon (2013) also recommends developing layers of information that are accessed via *progressive disclosure*. Progressive disclosure is a technique that allows users to see a few, important options but also have access to more details at their request. Carey et al. (2014) emphasize how progressive disclosure is key to providing effective documentation. By moving information that is advanced or seldom used to secondary screens, users can learn applications more quickly and make fewer errors. Furthermore, information can be progressively disclosed by placing it in small chunks where the user is most likely to use it. Field labels, messages, page descriptions, and hover text are all examples of how information can be disclosed as it is needed.

Layering

Silveira, Barbosa, and de Souza (2004) promoted the idea of layering information by *levels of affordances*, which classifies information based on questions users ask into three categories. The *operational category* answers the question "What's this?" and provides information users need to know to continue working with an application. The *tactic category* is related to completing a task and answers questions such as "How do I do this?" The *strategic category* addresses concepts, decision making, and problem solving questions such as "Why does this do this?" Depending on the question and the category of affordance, different types of user assistance are recommended. For example, when a user asks "What's this?" regarding an

element on the user interface, the first information presented should describe the item (operational). If the user wants more information on how the item is used, additional information is presented to explain the item's usage (tactical). If the information presented is short enough, both operational and tactical information can be presented at the same time.

Modern User Assistance Solutions

User assistance is a form of communication. Pierce (2012) believes that communication is information that is designed for the user. "I believe that design of communication is THE weak point of information today. If I were to summarize my criticism of information technology, I would say that information is useful only if you can get to it easily" (Pierce, 2012, p. 35).

Today, after research, experimentation, and results of usability studies, a set of user assistance techniques are available that can be used in a variety of software situations. These techniques allow users to access the information they need quickly, often without even realizing they are accessing a type of help system. Ellison (2007) explains that the advantages of using embedded user assistance include the ability to offer help at the precise location where the user needs help ("points of pain"), it helps to keep users working on the task they are doing, it can provide links to additional information, and users see the embedded user assistance as part of the application instead of an external help system.

Tidwell (2011) recommends providing the following types of user assistance depending on user needs.

Captions. Use short, concise verbiage to name fields. You can also add brief text under input fields to describe what users should enter.

Tool tips. Briefly describe interface features in a pop-up window. The window pops up when the user points at an interface feature such as a button or tool.

Hover tools. Similar to tool tips, hover tools display text in a pop up bubble when the user points at something on the interface.

Longer help text in collapsible panels. If there is space on the application window, a panel can be added to display help. The help changes as the user displays another tab or portion of the window.

Introductory screens. When a user starts an application for the first time, an introductory screen can appear to orient the user toward the first steps to take. The introductory screen may also include links to longer help topics.

Online manual. Online manuals and online help systems can be accessed from a link or button on the application.

User Assistance Recommendations for eLearning

Alessi and Trollip (2001) provide a thorough set of guidelines and recommendations for including user assistance in multimedia learning applications. These recommendations encompass the best practices regarding minimalism, progressive disclosure, and layering and include some of the modern user assistance techniques recommended by Tidwell.

Procedural vs. informational help. *Procedural help* explains how to use the eLearning program and it should be available to users at all times. *Informational help* is assistance provided for using the content of the eLearning program. For example, you can include examples, sample problems, or glossaries to provide help with the content of the eLearning program. Not all eLearning programs need informational help and it is more often used with tutorials than with exams or drills.

Novice vs. experienced users. It is important to understand that "first-time users have different needs and expectations than repeat users of a program" (Allesi & Trollip, 2001, p. 50).

To service this need, only give basic information such as how to operate a computer to learners that you know are novice computer users.

Operational procedures. At the beginning of the eLearning program, give basic instructions on how to use the software, such as how to navigate from place to place. Do not provide detailed instructions for portions of the program that are used later. For example, do not provide instructions for taking a quiz at the beginning of the eLearning program.

Basic directions. Provide basic directions at the beginning of the eLearning program that are brief and provide an appropriate level of detail. Start with minimal instruction and add detail as needed that is revealed through usability testing. Provide detailed instructions only when needed, such as if you include a complicated simulation in the eLearning program. Throughout the program, provide an easy way to return to the Directions page such as with a button or a menu option.

Multimedia features. If you include videos or audio information in the eLearning program, include brief instructions on how to use basic features such as pausing or stopping.

Context sensitive text. Context-sensitive user assistance is help that is provided precisely where it is needed. For example, a rollover can provide text that explains a button on the screen or a graphical component of a simulation. Rollovers are easy to implement and ensure that help is always available to the learner. Provide informational help as rollovers whenever possible.

Help button. Make sure a Help button is available to the users at all times. It can direct the learner back to a Directions page, or to a page that offers the appropriate assistance.

Printed or external material. Help can also be provided as a printed or electronic manual. These variations are suitable for when the learner needs to understand something before starting the eLearning program, such as how to launch the program.

Usability testing. Often, learners need information that the designer or developer cannot foresee. It is wise to perform usability testing to identify areas where user assistance methods can improve the learner's understanding of how to use the program.

Conclusion

User assistance emerged in 1988 in the form of an external online help system that was linked to a DOS application. Over the years, several standard types of user assistance were developed to help users find information quickly, without having to leave the software application they were using. These user assistance techniques allow users to access just the information they need, when they need it and incorporated the concepts of minimalism, progressive disclosure, and layering. Most user assistance techniques can be adapted into eLearning programs to provide the same type of help to users.

References

Alessi, Stephen M. & Trollip, Stanley R. (2001). *Multimedia for Learning: Methods and Development* (3rd ed.). Needham Heights: Pearson Education.

Carey, M., McFadden Lanyi, M., Longo, D., Radzinski, E., Rouiller, S., & Wilde, E. (2014). *Developing quality technical information: A handbook for writers and editors*. Upper Saddle River, NJ: IBM Press.

Ellison, M. (2007). Embedded user assistance: The future for software help? *Interactions*, 14(1), 30.

Farkas, D., & Williams, T. (1990). John Carroll's the nurnberg funnel and minimalist documentation. *IEEE Transactions on Professional Communication, 33*(4), 182-187.

Gallon, R. (2013). *A cognitive design for user assistance: Users become learners* [PowerPoint slides]. Retrieved from http://www.slideshare.net/Culturecom/a-cognitive-design-for-user-assistance-users-become-learners.

Miller, B., & McCandless, P. (1999). Online help supports performance. *Performance Improvement,* 38(4), 32-35.

Pierce, R. (2012). Design of communication. *Communication. Design Quarterly Review, 1*(1), 31-36.

Self, T., & Bleiel, N. (2010, November). The history and future of embedded user assistance. *Tcworld magazine for international information management.* Retrieved from http://www.tcworld.info/e-magazine/content-strategies/article/the-history-and-future-of-embedded-user-assistance/.

Silveira, M. , Barbosa, S. , & Souza, C. (2004). Designing online help systems for reflective users. *Journal of the Brazilian Computer Society, 9*(3).

Stahl, E., & Bromme, R. (2009). Not everybody needs help to seek help: Surprising effects of metacognitive instructions to foster help-seeking in an online-learning environment. *Computers & Education, 53*(4), 1020-1028.

Tidwell, J. (2011). *Designing interfaces* (2nd ed.). Sebastopol, CA: O'Reilly.