VOLUME 17
NUMBER 2
May 1993

THE JOURNAL OF COMPUTER DOCUMENTATION

1 **Introduction to First New Article Issue**
by Russell Borland, Editor-in-Chief.

3 **The Role of Balloon Help**
by David K. Farkas. Sees Balloon Help as a successful implementation of minimalist principles that nevertheless needs to be supplemented by other forms of documentation.

20 **Comments on "The Role of Balloon Help" by David Farkas**
by John Talburt. Balloon Help does not fit the true model of minimalism, but does represent an initial step along the path toward the re-integration of user documentation and system design.

21 **Comments on Balloon Help**
by Jonathan Price. Farkas' investigation is a first step toward re-evaluating Balloon Help; foresees further re-evaluation.

23 **Toward an Ethos of Rationale: Commentary on "The Role of Balloon Help" by David K. Farkas**
John M. Carroll. Sees Farkas' article as a reasoned analysis that draws upon the state of our knowledge to understand a piece of technology and further the development of it.

acm PRESS Association for Computing Machinery
1515 Broadway, New York, NY 10036
SIGDOC Special Interest Group for Documentation

The Role of Balloon Help

David K. Farkas
Department of Technical Communication
College of Engineering
University of Washington

Abstract. *Balloon Help, which is becoming standard in the Macintosh world, enables the user to display brief annotations of interface objects by passing the pointer (cursor) over those objects. This investigation explains the operation of Balloon Help, presents the theoretical and empirical rationale for Balloon Help, assesses its value in supporting both exploration of an interface and task-focused behavior, considers its relationship with other forms of help, and evaluates some possible modifications of Balloon Help. Balloon Help is viewed as a successful implementation of minimalist principles that nevertheless needs to be supplemented by other forms of documentation.*

Balloon Help was introduced into the Macintosh operating system with the System 7 release. Balloon Help was used by Apple in documenting System 7, and Apple strongly supports its use by all developers of software for the Macintosh. A great many developers are incorporating Balloon Help as they introduce new products and recode their existing products for System 7, and so Balloon Help will very likely become standard in the Mac world (Gassée, 1991).

In this study, I analyze and assess Balloon Help. Specifically, I

- explain its operation,
- present it as a form of interface annotation incorporating minimalist principles,
- assess its effectiveness when used both for initial familiarization with a product and for accomplishing tasks,
- consider its relationship to and potential duplication of other forms of online help, and
- evaluate some possible enhancements of Balloon Help.

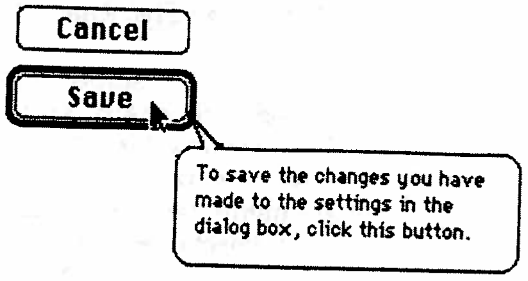
Balloon Help has received generally favorable commentary in the trade press (Swaine, 1990; Matthies, 1991; Poole, 1991; Davis, 1991); it is applauded for providing users with quick, convenient access to help information. But the reviews are not consistently positive. One commentator calls Balloon Help "little more than a gimmick" (Reed, 1991), while another refers to it as "something my 5-year old child needed occasionally" (Levitan, 1991). In my own experience, Balloon Help has left many experienced Mac users unimpressed. It is not, after all, a stunning technological advance: It employs no AI technology, interactive dialog with the user, multimedia, or sophisticated hypertext linking. Much flashier online aids have appeared of late. Furthermore, if a help system is measured by the amount of information it delivers to the user, Balloon Help is certainly less than impressive. Despite all the things it is not, Balloon Help, I maintain, is effective in a variety of situations. Simple and undramatic, it is an instance of appropriate technology in the world of online assistance.

How Balloon Help Works

Users enter the Balloon Help mode by selecting the Show Balloons command from the Help menu on the Macintosh menu bar. They exit this mode by means of the Hide Balloons command on the same menu. Once in the Balloon Help mode, many interface objects on the screen are "hot." That is, they will display small "balloons" (see Figure 1) containing brief help messages when the user moves the pointer (cursor) over them. These balloons, which are named after the balloons used in comic book dialog, appear at the location of the hot spot. Each balloon has a "tip" that points precisely at the hot interface object. Balloons are parsimonious in design: there are no buttons to click, no scroll bars, no title area. The dimensions of the balloon are barely larger than the space required for the balloon message.

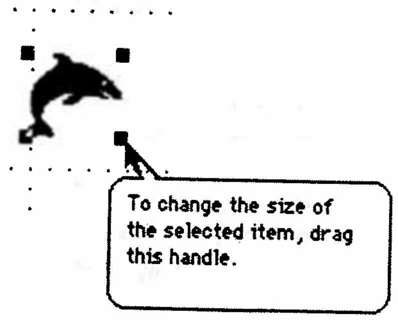
Which spots will generate balloons? This depends on the software developer. Balloons for the Title bar, Close box, and other unvarying Macintosh interface objects are provided by System 7. But software developers can put balloons almost anywhere, and the process is relatively easy, especially with the BalloonWriter utility. Even temporary interface objects like handles for graphics can trigger balloons. Furthermore, a different message can be written for the same spot when it is in a different condition. That is, an option button (radio button) may display a different message when it is selected, unselected or unavailable for selection (dimmed). Other than this moderate degree of context sensitivity, the display of balloons is unrelated to deeper-level changes in the system's state or the actions of the user; Balloon Help is not intelligent online assistance. For both technical and communicative reasons balloon messages must be short, and although graphics can be

Balloon for a default Save button

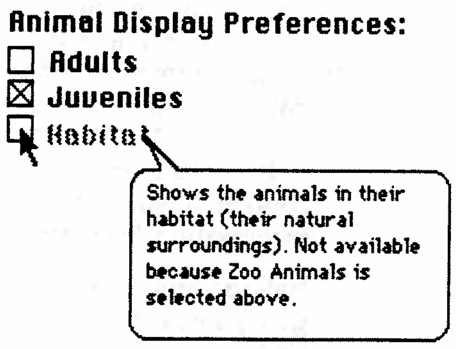


To save the changes you have made to the settings in the dialog box, click this button.

Balloon for a selection handle



To change the size of the selected item, drag this handle.



Animal Display Preferences:

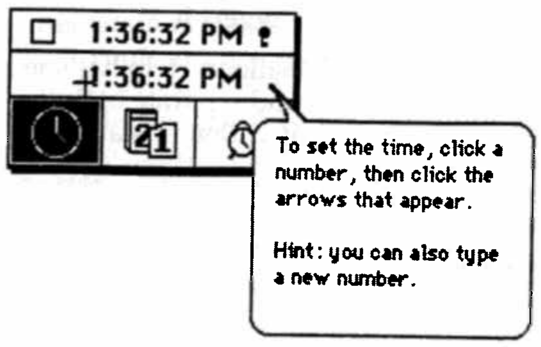
☐ Adults

☒ Juveniles

☐ Habitat

Shows the animals in their habitat (their natural surroundings). Not available because Zoo Animals is selected above.

Balloon for setting a clock



To set the time, click a number, then click the arrows that appear.

Hint: you can also type a new number.

Figure 1. Several examples of Balloon Help on display.

placed in balloons, this is rarely done. Typically, the messages explain the purpose of the interface object (*Apple Publications Style Guide*, 1991).

A balloon is dismissed as soon as the pointer leaves the hot spot; thus only one balloon at a time is displayed. Even so, an often-cited problem is "balloon barrage." Users are distracted by numerous balloons that appear unintentionally as the user moves the pointer toward the next object of interest. To limit the number of unintentional balloons, the pointer must pause for 1/10 second over a hot spot before a balloon is triggered (Matthies, 1991). Consequently, if the user moves the pointer decisively from one location to the next, unintended balloons will not appear. Computer users, however, do not necessarily move the pointer decisively, especially when they are looking for information, and so balloon barrage remains an issue.

One solution for balloon barrage has been offered by the developers of "init" utilities, such as Helium, that enable the user to toggle in and out of Balloon Help mode from the keyboard. In the case of Helium, Balloon Help is active only while a key combination is held down. This quick-toggle feature permits the highly selective display of balloons, but eliminates the automatic, effortless quality of the original implementation. Summing up the salient features of Balloon Help, we can say:

1. Balloon Help is spot-triggered. Unlike many forms of context-sensitive help, a developer can trigger help information from almost any pixel on the screen.
2. Balloon Help is spot-displayed. That is, in contrast to many forms of context-sensitive help, balloons appear right next to the object that triggered them. Because of spot-display and also because of the balloon tip and the small size of balloons, there is a close association in the user's mind between a balloon and the object that triggers it.
3. Balloon Help has some awareness of the system state and separate balloons can be written for the same object in different states.

One way is to point out a limitation of the now-dominant graphical-user interface (GUI): the numerous graphical controls and icons they contain are not completely intuitive and self-disclosing, and there is rarely enough screen "real estate" to explain everything users need to know about the system. New users of a GUI do not reliably infer the function of such standard objects as scroll bars and zoom boxes, and even users who are experienced with a particular GUI cannot predictably infer the function of the various application-specific icons and other graphical objects they encounter in an unfamiliar product. In addition to mysterious graphical objects, graphical-user interfaces contain many text labels, such as command names and labeled check boxes and option buttons. Because of space constraints, these labels are often too brief to be meaningful. Non-graphical interfaces face much the same problem, but there is likely to be a higher proportion of brief text labels to graphical objects. Balloons, then, can be thought of as interface annotations, elaborative comments. But whereas permanently displayed screen annotations, "persistent help" in Kearsley's terms (1988), would hopelessly clutter the screen if they were placed everywhere the user could benefit from them, balloon annotations appear when they are needed and disappear when they are not.

A second rationale can be drawn from the theoretical and empirical work conducted for IBM by John Carroll (Carroll, Smith-Kerker, Ford, and Mazur-Rimetz, 1987-88; Carroll and Rosson, 1987; Carroll, 1990). This work defined and popularized the concept of minimalist documentation. Carroll made the important observation that computer users are impatient and highly curious and that, rather than reading extended documentation, they want to begin immediately working with the product (Carroll, Smith-Kerker, Ford, and Mazur-Rimetz, 1987-88; Carroll and Rosson, 1987). Carroll also observed that, even while they are just starting to learn a system, users want to get actual work done, and—again—that users prefer to bypass documentation or use the briefest possible documentation as they accomplish their work (Carroll and Rosson, 1987). The minimal manual was Carroll's primary effort to satisfy these desires of computer users (Carroll, Smith-Kerker, Ford, and Mazur-Rimetz, 1987-88). Balloon Help can be seen as an online implementation of the original minimalist idea. Users forego introductions, conceptual overviews, any instructional curriculum, and complete procedures for quick access to explanations and hints that will support their own explorations and task-focused efforts with the software.

Balloon Help for Tasks

While important, familiarization is only one part of a user's life history with a software product. Users, as noted above, are curious and wish to explore, but they also have a very strong urge to accomplish actual work. How well, then, does Balloon Help support users when they have completed initial familiarization and are seeking online help in support of real tasks?

Finding the Right Balloon

Information access is not an issue when users are exploring an interface to gain familiarity. The user sees an object, wonders about it, moves the pointer, and views a balloon. But once the user has committed to trying to accomplish a particular task, information access becomes paramount: users must identify the appropriate interface objects before they can access the relevant balloon. Balloon Help, therefore, requires the user to draw inferfences from the interface. Carroll applauds documentation that encourages this kind of active learning as well as documentation that keeps the user's focus on the working interface rather than on pages of a manual or windows of help information.

The success of this problem-solving activity, however, depends both on a particular user's skill at inferring and the quality of the interface. If the key control for the desired task is buried four levels deep in the interface or is placed under an unlikely menu, the user might never find the requisite control and its balloon. There are also procedures that are not associated with any particular part of the interface, leaving the help writer with no good place to associate a balloon. On the other hand, if the interface is well designed, information access via Balloon Help is likely to be fast and accurate. A prototypical instance is the user who finds the command that seems to match the task goal, uses the balloon to confirm that choice, displays the dialog box for that command, and then uses the dialog box balloons to provide convenient capsule explanations of the dialog box options.

Following the Procedure

Balloons are necessarily brief. As Sellen and Nicol point out, balloons often describe the function of an interface object rather than present a series of steps that will encompass the complete task. In the case of dialog box options, this is no limitation, because the dialog box option is, in effect, a self-contained mini-procedure that enables a user to complete a task in the exact manner the user desires (e.g., printing, but printing only a portion of the document).

If, on the other hand, the user selects a block of text, displays the balloon for the Copy command, and reads that the copy command "copies the selected text and graphics to the Clipboard," the user must be able to complete the task (pasting the selected text or graphic back into the document) from other knowledge of the system. Alternatively, the help writer may write a longer balloon that includes an explanation of pasting, thus relieving the user of this burden. The sample balloon for setting the time is interesting and impressive because in only 21 words it packs a purpose statement (to set the time), and three action steps (clicking a number, clicking the arrows, and the alternative method, typing a number), along with a feedback step (arrows will appear).

Many products, however, require much lengthier and conceptually more complex procedures in which several steps are decision points (conditionals) for which guidelines must be provided. In these instances, the inability of balloons to provide more than the briefest conceptual information and feedback information becomes a limitation. Also, many complex procedures require users to operate controls located in disparate parts of the interface. Few users will be willing to successively consult a series of balloons as they carry out a single procedure.

Clearly, then, both in terms of information access and presentation, there will be products and portions of products in which the practical limits of Balloon Help are exceeded. Both Sellen and Nicol and the *Apple Publications Style Guide* acknowledge this fact. On the other hand, Balloon Help is an excellent means of providing familiarity and supports task-focused behavior over a broad range of product functionality.

Complementing Balloon Help

If Balloon Help does not fully support task-focused behavior, a good means of complementing Balloon Help is not far to seek. One of the most prevalent forms of help consists of windows or panels of help information accessed by a hierarchy of descriptive phrases. The user scans a menu or some other listing of top-level entries and then navigates down into a hierarchy of more specific entries until finding the title of the desired procedure. (Other hierarchies can be devised for commands, keyboard shortcuts, etc.) Accessing windows or panels of procedural information in this way is comparable to using the table of contents in a printed user's guide. Constructing an effective procedure hierarchy requires a systematic analysis of the tasks the user might want to perform and a mapping of these tasks to the functionality of the product. But if this is done correctly, the access is relatively immune to quirks in the interface, and supports users who do not want to explore an interface and infer which objects support which tasks. These users only deal with the interface when they follow instructions for carrying out a procedure.

Another traditional form of access is the keyword list or online index, the online equivalents of the traditional back-of-book index. Here the help writer complies an extensive alphabetical listing of words and phrases that are meant to correspond to the phrases that users are likely to formulate to represent their goals. In both cases, the user does not find help information from the working interface, but rather consults listings of task-oriented phrases devised by the help writer. So, a good complement to the interface-based access provided by Balloon Help (and various other forms of context-sensitive help) is its diametric opposite, what we can call "phrase-based" access to help information.

Apart from information access, another reason why phrased-based access is a good complement to Balloon Help is information presentation. In almost all implementations, phrase-based access provides much more complete help information than does Balloon Help. Typically, the user is shown a full window or panel from a library of help topics, and this window (or panel) can offer scrolling or paging through the help topic as well as links to other help topics in the help library provided by a browse sequence, hypertext jumps, and pop-up definitions. Also, because these windows are not associated with particular interface objects and, in the better implementations, remain on the screen while the user works with the product, they are well suited for documenting procedures that involve disparate parts of the interface.

Balloon Help and System Prompts: Is There Duplication?

Although phrase-based access to detailed help information complements Balloon Help, there might well be significant duplication of function if a software product included both Balloon Help and some other form of help optimized to display brief help messages.

There is a form of help, in fact, which has been implemented along with Balloon Help in certain products, which in some respects resembles Balloon Help, and which might well be perceived as duplication of Balloon Help. This is help in the form of system prompts. It is worthwhile, therefore, to clarify the relationship between system prompts and balloons and to demonstrate that any duplication is incidental and a kind of historical anomaly.

Many computer products offer system-initiated messages of various kinds. These include error messages, alerts of important actions (Do you wish to overwrite the file: Lovenote), progress and completion messages (Backing up the file: Lovenote . . . Backup completed), and prompts for the next user action.

Error messages and alert messages usually appear prominently on the screen and require some explicit action before the user can resume normal operations with the software. Other system messages, including prompts, typically appear in a small message area located at the bottom or some other margin of the screen and do not require any explicit response. The user simply continues working with the product and can either heed or ignore this information.

In current systems, prompt messages are sometimes similar and even identical to balloon messages. Sun Microsystem's *Open Look UI Style Guide* (1989), for example, suggests the following prompt when the user has selected the rectangle tool from a drawing palette: "Position pointer then drag—Rectangle Tool." A balloon for the rectangle tool might be quite similar. Moreover, in some Microsoft products, certain balloons are similar or identical to prompts that appear in the status line located at the bottom of the screen. If a software product offers two forms of help that might display the same message, is there a significant overlap in function?

One difference, of course, is that prompts are not spot-displayed. Furthermore, whereas balloons are sensitive to the position of the pointer over an object, prompts require the object to be selected or given some explicit focus. This difference has the practical result of limiting the range of objects for which prompts can be written and in making access to prompts slower than access to balloons. But this difference also points to a more fundamental difference between Balloon Help and system prompts, the difference in their essential nature and ultimate evolution. Balloons are annotations and explain the purpose of interface objects. Prompts are directive in nature; they reflect the system's record of the user's recent actions and best guess as to the user's current intentions. In intent, they are not explanations of interface objects but explicit instructions for what to do next.

Currently, prompts can provide explicit instructions only in highly restricted or highly structured domains such as automated bank tellers, simple e-mail systems, logon procedures for mainframes, and data-entry screens. In more complex domains, however, it is often impossible to effectively track and anticipate user actions, and so help writers often can do no more than write balloon-like annotations explaining the function of the most recently selected object. The prompt, then, becomes no more than a hint and a somewhat inferior form of Balloon Help.

But despite the current limitations of prompts in complex domains, computers will achieve intelligent prompting. They will track more user actions, make better inferences about these actions, query users for clarification of their intentions, and offer more detailed advice that can include conditional instructions (If you are trying to do A, then....) Thus, as prompting becomes more intelligent, there will be increasing divergence in the nature of prompt messages and balloon messages.

Modifying Balloon Help

What is the potential for modifying Balloon Help and for creating new, more refined help engines on the general model of Balloon Help? What kinds of changes are worthwhile enhancements? Should changes be implemented that alter the fundamental character of Balloon Help?

A common thread running through all these potential modifications is the issue of added complexity, either in the way in which the user operates Balloon Help or in the nature of the help display. Complexity is the bane of help systems, and so careful thought and usability testing are necessary to confirm the value of any potential modification.

Quick Toggling

The prevalence of Helium indicates user interest in a quick-toggle feature that permits balloons to be accessed deliberately. The choice between system-initiated and user-initiated display of balloons is highly individual, but a means of preventing unwanted balloons would be particularly valuable for users engaged in task-focused activity.

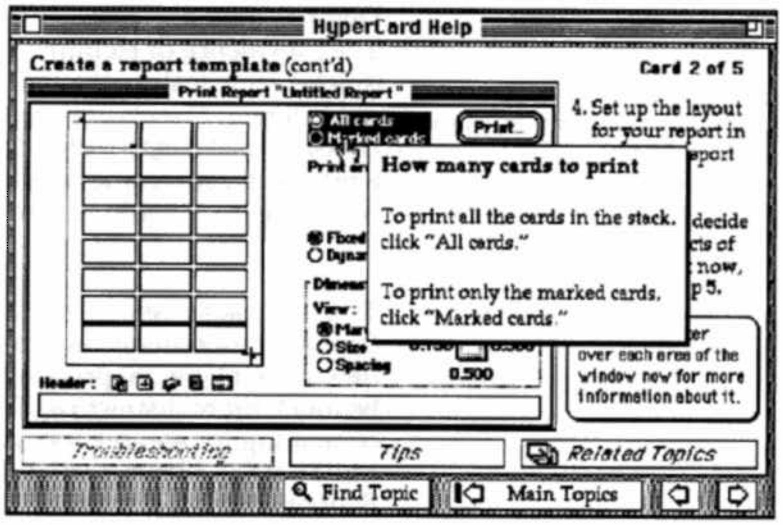
This is because task-focused activity, much more than familiarization, requires problem-solving and other deeper-level mental processes, and so balloon barrage can be much more distracting. The proliferation of balloons is competing for scarce processing resources (Navon and Miller, 1987). I have personally seen several users turn off Balloon Help as they made the transition from familiarization to trying to accomplish a new task. Had Helium been installed, they might well have continued to use Balloon Help. It therefore seems that with some sort of Helium-like quick toggle, a major impediment for using Balloon Help to support task-focused behavior is removed.

Filtering System-Level Balloons

A problem related to balloon barrage is that users who have learned the basics of the Macintosh interface soon tire of repeatedly viewing the balloons for standard and very familiar objects such as the title bar, scroll bar, and inactive windows. A possible modification of Balloon Help, therefore, is a filtering option that would eliminate the underlying System 7 "beginner's balloons" and retain only balloons specific to the application being used. The principle of "layering" information for different users is central to documentation, and this modification is an implementation of this principle.

Revealing the Area Triggering a Balloon

Users can benefit from an immediate visual cue indicating the number of interface objects a particular balloon pertains to. For example, the use of highlighting could more strongly distinguish an instance in which one balloon pertains collectively to three related option buttons from an instance in which three related option buttons have three separate balloons. This concept was implemented in the precursor to Balloon Help that appears in some HyperCard 2.0 help panels. When the balloon is displayed, the entire region that triggered the balloon is highlighted until the user moves the pointer out of this region and dismisses the balloon (Figure 2). Although this feature adds visual complexity to balloon display, it might be worth implementing.



How many cards to print

To print all the cards in the stack, click "All cards."

To print only the marked cards, click "Marked cards."

4. Set up the layout for your report in

over each area of the window now for more information about it.

Figure 2. HyperCard 2.0 help screen showing highlight for the area that triggered a "balloon."

Links to the Standard Help Library

Many software products offer context-sensitive access to the standard help library normally accessed by topic hierarchies or keywords. In one implementation the user selects a particular interface object and presses a key; in another, the user turns the pointer into a special help pointer and then selects an object. This form of help supports interface-oriented problem-solving, somewhat as Balloon Help does, but provides detailed information rather than brief balloon messages.

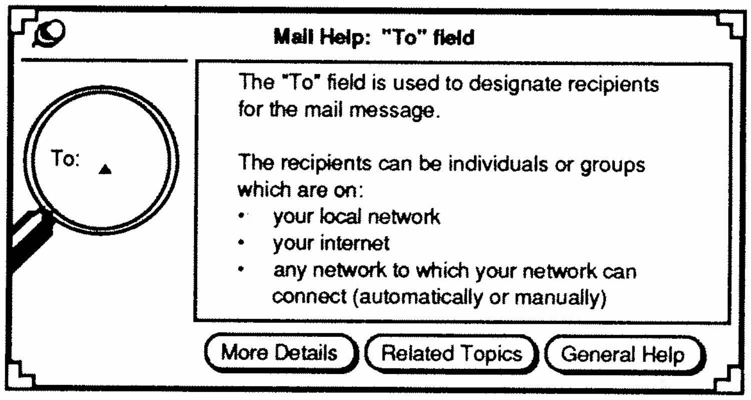
A possible enhancement to Balloon Help is to provide rapid access from any balloon to the most appropriate screen in the standard help library. Balloons might have their own hot spots (a bit tricky to implement) or the F1 key could be used. The help windows in Sun's OpenWindows Spot Help includes a button that brings up the standard help reference, Help Viewer, displaying a list of topics generally related to the topic of the Spot Help window. The *Open Look UI Style Guide* (see Figure 3) suggests a more elaborate variation in which the help information window contains three buttons for accessing more help information.

Graphics and Multimedia

At a time when multimedia help is appearing, Balloon Help does not even utilize graphics. Should something be done? In many instances, Balloon Help does quite well without graphics. The most common form of graphic in computer documentation is the screen representation that shows a specific portion of the interface to the user. Balloon Help uses its own association with the relevant object as a very adequate substitute for screen representations. While animated documentation is not always the best means of providing help information (Palmiter, Elkerton, and Baggett, 1991), animation is often highly desirable for explaining processes and other documentation tasks. A company called Motion Works has developed a product that software developers can use to create special balloons containing brief animated sequences.

More "Intelligent" Balloons

An interesting issue is the desirability of providing greater context sensitivity and even "intelligence" for Balloon Help. There are clearly benefits in adding greater context sensitivity to Balloon Help. For example, in a word processing program, the command for adding footnotes might trigger separate balloons depending on whether footnotes had already been created for the document. Going further, the way in which footnotes had been added to the document could dictate the nature of the balloon message. But even if greater context sensitivity is added to Balloon Help, its fundamental character should not be changed. It should remain annotative and descriptive, an aid to users as they figure out what to do, and not a means of providing (or trying to provide) explicit directions for the user's next action. Explicit directions are the natural evolution of prompts and other forms of help, such as alerts, which represent the best advice of the system as a whole and are not closely associated with specific objects on the interface. Furthermore, given limited resources, the large


Mail Help: "To" field

The "To" field is used to designate recipients for the mail message.

The recipients can be individuals or groups which are on:
- your local network
- your internet
- any network to which your network can connect (automatically or manually)

Figure 3. Buttons for immediate access to detailed help information.

investment entailed in developing intelligent help is probably best directed toward system prompts rather than Balloon Help.

Conclusion

This investigation of Balloon Help mentioned three broad families of help: context-sensitive help, help that uses phrased-based access, and help which takes the form of system prompts which track the user's interactions with the system and which aspire beyond context sensitivity toward true intelligence.

Balloon Help is a form of context-sensitive help featuring spot triggering of balloons, spot display of balloons, and balloons optimized for brief messages. Also, whereas most context-sensitive help is user initiated, Balloon Help, with the addition of Helium, can operate in either system-initiated mode (ideal for familiarization) and the more deliberate user-initiated mode (ideal for task-focused learning).

Balloon Help is brief and instantly available, and asks users to keep their attention on the interface and engage in inferential learning. Thus it broadly follows John Carroll's minimalist program, and is, in fact, a successful implementation of minimalism.

Balloon Help, nonetheless, has significant limitations, particularly in supporting long and complex procedures. For this reason, and because users should not be required to rely on the interface to find documentation for the tasks they want to accomplish, Balloon Help should not be the sole piece of documentation or even the sole piece of on-line documentation for a product.

Despite the mixed reception it has received, Balloon Help should have a bright future. The now-dominant graphical-user interfaces seem to sport ever more cryptic graphical objects, and commands have ever more options, all represented in dialog boxes (and other places) in very terse form. All this needs explication.

Another trend favors Balloon Help. As graphical interfaces become ever more prevalent and more standardized, users are becoming more familiar with such basic GUI operations as pulling down menus, clicking buttons, and typing into text boxes. Greater numbers of users, one might reason, will become impatient with step-by-step procedures that incorporate descriptions of these actions. Brief purpose-oriented statements, the ideal content of balloons, may be increasingly favored, and more users may migrate to Balloon Help from more conventional forms of documentation.

References

Apple Publications Style Guide. (Fall 1991). Cupertino, CA: Apple Computer, Inc.

BalloonWriter User's Guide. (1991). Cupertino, CA: Apple Computer, Inc.

Carroll, J.M. (1990). The Nürnberg Funnel: Designing Minimalist Instruction for Practical Computer Skill, Cambridge, MA: MIT Press.

Carroll, J.M., and Rosson, M.B. (1987). The paradox of the active user. In J.M. Carroll (ed.), Interfacing Thought: Cognitive Aspects of Human Computer Interaction , pp. 80-111. Cambridge, MA: MIT Press/Bradford Books.

Carroll, J.M., Smith-Kerker, P.A., Ford, J.R., and Mazur-Rimetz, S.A. (1987-88). The minimal manual. Human Computer Interaction. 3, pp. 123-153.

Davis, F. (May 20, 1991). Operating systems are evolutionary, not revolutionary. PC Week, 8, p. 165

Gassée, J.L. (August 6, 1991). System 7 getting pledge of allegiance. MacWeek 5, p. 64.

Hancheroff, M.C. (October 25, 1991). Personal Communication.

Levitan, A. (October 1991). A tale of two operating systems. Computer Shopper, 11, pp. 137 & 148.

Kearsley, G. (1988). Online Help Systems: Design and Implementation. Norwood, NJ: Ablex.

Matthies, K.W.G. (1991). Balloon Help takes off. MacUser, December 1991, pp. 241-48.

McClintock, M. (January 17, 1992). Personal Communication.

Navon, D., and Miller, J. (1987). Role of outcome conflict in dual-task interference. Journal of Experimental Psychology: Human Perception and Performance, 12, pp. 435-48.

Open Look UI Style Guide. (1989). Mountain View, CA: Sun Microsystems, Chapters 11 & 12.

Palmiter, S., Elkerton, J., and Baggett, P. (1991). Animated demonstrations vs. written instructions for procedural tasks: a preliminary investigation. International Journal of Man-Machine Studies, 34, pp. 678-701.

Poole, L. (July 1991). Confessions of a System 7 user. Mac World, 8, pp. 195-201.

Reed, S. (August 1991). Apple scouts ahead with System 7. PC Computing, 4, pp. 40-42.

Sellen, A., and Nicol, A. (1990). Building user-centered on-line help. In B. Laurel (ed.) The Art of Human-Computer Interface Design, pp. 143-153. Reading, MA: Addison-Wesley.

Swaine, M. (November 1990). System 7.0 watch: the read balloon. MacUser, 6, p. 244.

Thanks to Jean Farkas, Mark Hancheroff, Linda and David Leonard, Marshall McClintock, Maria Staaf, Kent Sullivan, and Jon Wiederspan.

Comments on "The Role of Balloon Help" by David Farkas

by John R. Talburt
Department of Computer and Information Science
University of Arkansas at Little Rock

My comments are aimed at two levels. The first is a direct response to Balloon Help as an online help facility, particularly from the standpoint of minimalism. The second is a more philosophical level relating to the changing face of computer documentation and the role of SIGDOC.

Professor Farkas best summarizes my own assessment of Balloon Help in his statement "Simple and undramatic, it is an instance of appropriate technology in the world of on-line assistance." In my view, Balloon Help is of more value for the problem that it addresses than its actual effectiveness as a help facility. My own experience, and that of every other user I spoke with, is that Balloon Help is a neat feature to demonstrate to someone else, but not something you use yourself.

So why does Balloon Help exit at all? Balloon Help and whatever intrinsic utility it may possess exist because intuitive graphic user interfaces are not entirely intuitive. A balloon explanation of the trash can icon is for those users for which the graphic has no operational association. The balloon explanation of the "duplicate" option of the "file" menu is for those users unfamiliar with (or confused about) the duplication operation and, consequently, are not cued by a single word description. In short, Balloon Help is intended to compensate for those failures of the graphic interface to allow the user to "recognize and select," rather than "recall and type." Despite its unobtrusiveness and spot sensitivity, Balloon Help is basically a patch on the window interface and, in my opinion, does not fit the true model of minimalism.

Before continuing with the minimalist aspects of Balloon Help, let me digress for a moment concerning "so-called" intuitive interfaces. I believe that intuitiveness comes more from standardization than from innate mental concepts. Automobiles are a case in point. A person with even limited driving experience can get into any automobile and immediately start driving. Starting and driving a car does not require a manual. Even though there have been many changes in automotive design, the artifacts of operation, such as the steering wheel, pedals, and light switches have become standardized over the years. A steering wheel seems an intuitive way to control direction now, but it is only one of many alternative designs that survived early experimentation, any one of which might now seem equally intuitive had it been adopted. Although software interfaces change much more rapidly and dramatically than automobiles, some aspects stay around long enough to become de facto standards. The example of the trash can icon above is so well accepted that a balloon description seems unnecessary except for completeness.

In my reading of the minimalist manifesto, *The Nürnberg Funnel* by John Carroll, I see the paradigm for minimalist documentation as an "expert user sitting at the next desk." A person who is there when you need them, but not always looking over your shoulder. Someone who at a moments notice can look at your screen, enter a few keystrokes to extract you from a series of bad menu choices, and put you back on the right track.

Someone to whom you can explain the barest details of an application, and who can respond by presenting in simple terms (or even doing) the steps that will arrive at a solution. Calling on someone already well versed in an application is certainly the most effective, and perhaps the most common, form of online help available today.

What Carroll is really telling us is what every technical writer has already experienced. There are so many degrees of freedom in audience requirements that every manual, no matter how well designed and written, will always fail deliver all that is demanded of it. User documentation is not about writing, it is really about the computer-human interface. Although writing has played a major role in the past, it is only one medium of communication, and in the broader understanding of documentation as user support, a role that is decreasing. Just as computer science struggles with the erroneous perception that "computing equals programming," in documentation it is "user documentation equals writing."

The present transition from paper documentation to online documentation and graphic user interfaces is only the first of many steps that will lead to the expert-at-the-next-desk interface. Written user documentation will continue to be replaced by more intelligent system interfaces employing other communication media. Although they do not yet exist, the models for such interfaces are already before us. From HAL and R2D2 to Apple's Navigator to Star Trek and Mr. Data, futuristic computer systems all have expert-at-the-next-desk interfaces. Each one shares two key features that will be part of the software, oral natural language communication and completely non-procedural operation. Each one passes the Turing Test summa cum laude.

Balloon Help to Star Trek may seem to be a non sequitur, but the point is that the reintegration of user documentation and system design is well underway. More and more, documentation and user support will reside within the application system itself. From a computing standpoint, "everything that can be automated, will be," and documentation is no exception to this rule. For SIGDOC this is merely completing a cycle beginning with our roots in system (logic) documentation. As a part of the Association for Computing Machinery, an organization primarily concerned with computing, SIGDOC is in an enviable position to lead in the transition to computer-based documentation. I hope it will do so.

Comments on Balloon Help

Jonathan Price
President, The Communication Circle
Albuquerque, NM

In his excellent article, David Farkas points out several ways in which Balloon Help can encourage absolute beginners to explore and become familiar with the basics of the Macintosh interface. Too many of us who are intermediate and expert users dismiss Balloon Help as a joke because it soon gets in the way, it is stupid about our intentions, and it rarely answers task-oriented questions, despite the creators' obvious attempts to weave procedural instructions into the definitions.

We may have been too hasty. The first Balloon Help described only a limited portion of the system software. Now we have a second generation: balloons explaining the detail of full and complex applications. When software vendors take the time to waft balloons above every item in their interface, including the options on the smallest and least important dialog box, they provide intermediate and experienced users with a great new tool, one that even Professor Farkas may have hesitated to predict.

Let me give you a personal example. I recently bought an update to More 3, an outlining program I have used intensively for years. I probably qualify as a fairly savvy intermediate user. With this update came Balloon Help, and when I turned it on, for the heck of it, not expecting more than the tidbits I'd seen from Apple, I was surprised to find that in fact I was learning a lot, such as:

- **What else you can do.** I moved over the current time blinking in an area called the status bar, and the balloon arose, and told me, "Press to display a menu of status items." I pressed, and saw the menu. I found that I can switch to a label that tells me how deep I've waded into my outline. A small, but very useful discovery.
- **What distinguishes two similar icons.** In the past I've used the tree chart so rarely that I got its icon confused with the lookalike icon for the bullet chart view. The balloons set me straight.
- **What I never noticed.** The updated ruler contained a new box, containing tiny numbers. I hadn't paid it any attention, until the balloon said that this was a popup menu, in which I could select a type of label for each level of headline without going through the sometimes six steps or more to bring up and use the relevant dialog box. That discovery alone has saved me enough time to justify fooling around with the balloons.
- **What I always had to try out to understand.** The application retains some of the ambiguous commands of its youth; for instance, the Library menu has a command Outlines that leads to a submenu beginning with the command Install Template. Now what does that mean? If I install a template, will I wipe out my current text, or its format? Balloon Help says: "Lets you add the current outline to the open library." What a relief! I may upset the library, but not my current work. The second command is: Remove template. Which one? With what results? Balloon Help says: "Lets you delete templates from the open library." Thank goodness: we're just talking about subtracting templates from the library. And what exactly is a template? Some other commands on the menu are Address File Template, Fax Cover, Incoming Calls Logs; for each of these commands, Balloon Help says, "Lets you paste this outline template in your current document." So I conclude that these templates include some prepared text, formatted to let me record phone calls, or make a cover sheet for a fax.

You can see that although I've have worn down the grass on several trails through this software, I haven't strayed far from my immediate work needs. Balloon Help has made exploring more rewarding, and taken some of the fear out of trying unfamiliar commands, when I'm taking a little break from work. (In the past, I've chosen a wrong button, and gone into some world I never knew, from which I only returned by accident, after half an hour of increasing despair.)

I predict that Balloon Help will recover from its current disrepute when intermediate and expert users find their applications alive with these not-so-annoying invitations to learn. Professor Farkas' excellent survey of Balloon Help's pluses and minuses lays the groundwork for such a reassessment; but the burden of proof lies with the technical writers and instructional designers who are now developing armadas of balloons for individual applications.

Toward an Ethos of Rationale: Commentary on "The Role of Balloon Help" by David K. Farkas

John M. Carroll
IBM T. J. Watson Research Center
Yorktown Heights, NY 10598

Professor Farkas' article address itself to a single interface technique. As he makes clear, the Balloon Help technique fills a fairly narrow role, which is nonetheless an interesting and potentially important role. The plan of the article is to first briefly explain what the technique is, then analyze its rationale in terms of theories and facts about users and documentation. Next, the article focuses down on some characteristic situations of use, specializing the rationale and reflecting more specifically on potential problems and strengths of the technique in these usage contexts. Finally, it turns to contrasts and conflicts of the technique with related techniques, and possibilities for extending the technique.

Such a discussion is quite useful in itself: Balloon Help is a current idea, much discussed in the trade press. Having a bit of objective and thoughtful analysis ought to help us deploy it more judiciously. But beyond this, the article conjured a vision in my mind of a series of analyses of specific interface and information techniques—analyses of their psychological and social rationale, the contexts in which they are more and less appropriate, the extensions, customizations and near-neighbor techniques against which they must be evaluated and understood.

Professor Farkas did not invent Balloon Help; he is not selling it (in any sense). His article is not a trade press pundit's review, not is it a report of a narrow laboratory study. It is a reasoned analysis that draws upon the state of our knowledge to understand a piece of technology and further the development of it. I think we need more of this.

The point I make here is quite old. At the dawning of the seventeenth century, as part of his effort to establish practical and empirical science, Sir Francis Bacon called for the creation of a catalog of all human technology, every technique employed, every object fabricated, everything—along with its technical rationale. Bacon urged that with such a resource, the creation of new applications and new technology could be rendered more systematic, more cumulative, and more predictable. It never happened.

For a couple of hundred years the idea limped along, occasionally trotted out in the charters of the various scientific societies. And to be sure, it was an idea easier suggested than implemented. But as modern "deductive" science took shape, Bacon's catalog was precisely the sort of footslogging empirical work routinely degraded and excluded.

I believe however that Bacon's approach may be just what we need. It's frequently an impossible leap to get from the contrived lab-paradigms of cognitive psychology to the rich usage situations we create, evaluate, and try to support. We need a "science base" organized around the things that matter in the real world we seek to change: tasks (like "following a procedure," in the Farkas article) and interface and information techniques (like Balloon Help).

This kind of science can be revelatory at just the right level. For example, Farkas reasons out a distinction between balloons and prompts that—I would claim—develops the state of our knowledge. Twenty-five semantic priming experiments and a whole boxcar of cognitive schemas would probably be less useful. I hope that more pieces like this one will appear in *The Journal of Computer Documentation*.