APPLIED RESEARCH

SUMMARY
- Hypothesizes that end users will use and find helpful any form of help that incorporates five key characteristics
- Describes two browser-based embedded help systems that incorporate these characteristics and presents results of a usability test

If We Build It, Will They Come? A Usability Test of Two Browser-based Embedded Help Systems

TREVOR GRAYLING

INTRODUCTION

MDL Information Systems is a supplier of chemical databases and associated database-searching software for PhD research chemists and biologists working in the pharmaceutical, biotech, and similar industries. The company's "core competency" is that chemists can draw a chemical structure in one of our chemical-drawing editors and then use this drawing (with additional data constraints, if required) as a query to a database.

The challenge for MDL's Technical Communications group is to enable our busy audience of chemists and biologists to figure out how to create effective and efficient search queries for the databases. A database may contain hundreds of thousands of chemical structures, one or more of which—if discovered in the database—could be worth millions of dollars to that company as a promising lead in developing a particular drug.

The big problem with database-searching applications is that the user receives little feedback. Consider, for example, novice users starting to use Microsoft Word. The users want to right-justify a paragraph of text. Their efforts, either successful or unsuccessful, will be immediately apparent on the screen: The paragraph is either correctly justified or it isn't. However, a good-quality or a poor-quality search query used over a large database may retrieve 5,000 records, whether good or poor. How is the chemist to know whether the search query was effective and efficient? That is, how does the chemist know that the search query retrieved *all* and *only* the relevant records?

Several years ago, we would have said that traditional online help (WinHelp invoked from the Help menu) was the solution: We thought that our rational and logical user chemists would be able to find all the user-assistance they needed at their fingertips to feel confident that they had created an effective and efficient search query. There would be no hunting for a manual, and WinHelp had good searching capabilities. Thus, for ISIS, our flagship database-searching application at the time, we carefully built a large, accurate, and comprehensive help system.

However, as noted in a previous article, "Fear and loathing of the Help menu," (Grayling 1998), things didn't turn out even remotely as we expected. Our users didn't behave in the way we anticipated, as summarized in the following section.

HOW END-USERS ACTUALLY BEHAVE

From our usability test with 10 customers and from similar tests of some of our other applications, as reported in "Fear and loathing," we found the following disturbing data:

- Users ignored the Help menu and used it only when truly desperate, having been told several times that the search query they had created for a particular scenario was incorrect.
- When they did reluctantly go to the help, they read the topics hastily and inaccurately. They were careless in choosing hyperlinks.
- They bailed out of the help system early, even without finding the needed information.

Although the user behavior noted in the second bullet above refers to users reading WinHelp topics, Steve Krug (2000, p. 21) notes that users behave very similarly when looking at Web pages: "[Users] *glance* at each new page, scan *some* of the text, and click on the first link that catches their interest or *vaguely* resembles the thing they're looking for. There are usually large parts of the page they don't even look at."

Manuscript received 10 August 2001; revised 8 November 2001; accepted 2 December 2001.

We did, however, notice one bright spot in the results of our testing: Users did use dialog-box help and "tooltips" help.

This data fitted well with reports from our Customer Service department, who reported that their phone representatives often act as an "online-help reading service" for user issues.

We received similar depressing data from Jared Spool's User Interface Engineering, a consulting company that has observed more than 2,000 end users testing a variety of user applications. They noted (Spool 1996, 1997) that

- Users overwhelmingly use trial and error.
- They don't go to help unless really stuck. When there, they browse hastily and often bail out even without the information.
- They don't use the help table of contents.
- What users *say* is not what they *do*: They may say, for example, that they will review overview material before using a product, but they don't.
- They don't do tutorials, but they do like self-contained examples.
- Users do try buttons labeled "Hints" or "Tips."

Usability guru Jakob Nielsen notes, "Nielsen's First Law of Computer Documentation is that *people don't read it*. This finding is even stronger for websites, where users truly shy away from any reading that is not essential to their task. Click on Help? Never. . . . Users read system documentation only when they are in trouble (that's the Second Law)" (Nielsen 2001a).

Since the "Fear and loathing" article was published in 1998, equally depressing additional data was made available at the 1999 and 2000 WinWriters Online Help Conferences.

DATA FROM THE WINWRITERS ONLINE-HELP CONFERENCES

In her 1999 presentation, Ginny Redish took an impromptu survey of the several hundred attendees at the talk. From a show of hands, it was clear, in the experience of the attendees, that end users:

- Use trial and error
- Dislike "help" (traditional help invoked from the Help menu)
- Are impatient
- Don't care about the product itself; they just want to get their work done
- Will repeat mistakes rather than go to help

In another 1999 presentation, Karen Schriver presented some results of a survey of 31 consultants and practitioners in our field. Concerning user attitudes, she noted that users

- Are even more impatient than before
- Will not read-to-learn

In his 2000 presentation, Michael Hughes noted the following from his own test observations:

- When trying to accomplish a task, reading procedures is (literally) the last thing users think about.
- Users go to any kind of help only when they are really stuck (and there is no one to ask).
- Users often read only the first few words of a procedure, exit help, and then wing it (again).

In another 2000 presentation, "Our customers hate using help" JoAnn Hackos noted that

- Some users don't know help exists.
- Other users know it exists but choose not to use it.
- Still other users use it, but get frustrated and then never return.
- In 14 site visits, she and her colleagues found no one using help.

"THE PARADOX OF THE ACTIVE USER"

As noted in "Fear and loathing," an explanation of this strange reaction to user-assistance information was first proposed in a profoundly provocative essay, "The paradox of the active user" (Carroll and Rosson 1987). In this essay, the authors note that "Most computer systems and their reference libraries [manuals, help] are designed with an inherently rational view of users in mind." However, they go on to plainly state that users' behavior appears to be far from rational, as summed up in two paradoxes (paraphrased here):

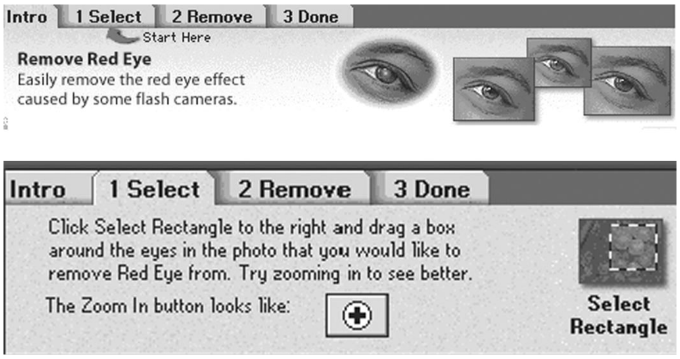
1. **The Production paradox:** Users are highly motivated to do their task but, paradoxically, are unwilling to learn the very software that would enable them to do their task effectively and efficiently.
2. **The Assimilation paradox:** Users bring their experience of other, similar software to using new software but, paradoxically, it often hinders more than it helps.

The result of the first paradox, the authors note, is that users then "asymptote [level out] at relative mediocrity." Or, as Krug (2000, p. 28) puts it, "Once we find something that works—no matter how badly—we tend not to look for a better way." This also explains why most self-identified "experts" that we test fare no better than novices in our usability tests.

We saw the effects of the second paradox at work during a usability test that we will discuss later in this article.

So what is the solution? This is a major problem for our field of technical communication. Early practitioners, writing for the user audience, first produced a variety of hard-copy manuals. When these were found to be little used, we moved on to online help (WinHelp invoked from the Help menu) as the great solution. We've now been exactly wrong on two occasions. We need to get it right—after all, at some point, our bosses are sure to notice!

To date, there *seem* to be two solutions: improving the user interface and using embedded help.



Remove Red Eye
Easily remove the red eye effect caused by some flash cameras.

Click Select Rectangle to the right and drag a box around the eyes in the photo that you would like to remove Red Eye from. Try zooming in to see better.
The Zoom In button looks like:

Figure 1a. Text in the user interface (Adobe PhotoDeluxe).

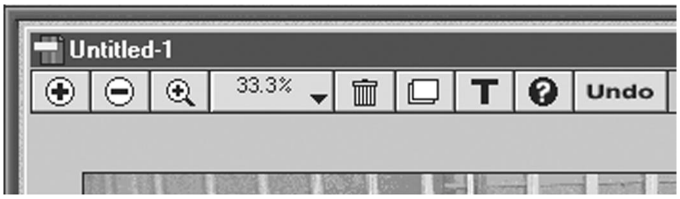
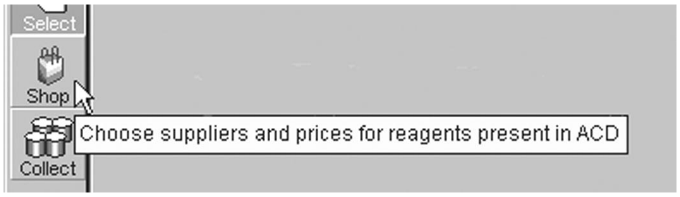
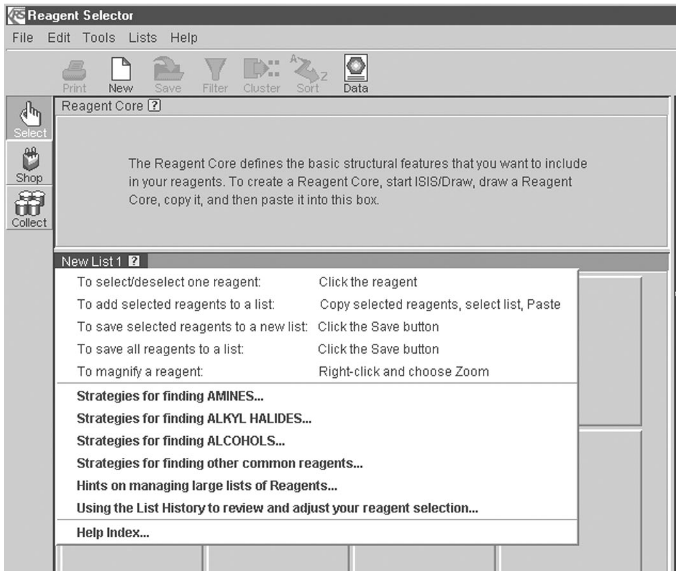


Figure 1b. Remove Red Eye procedure (Adobe PhotoDeluxe).



Choose suppliers and prices for reagents present in ACD

Figure 2. Tooltips (MDL's Reagent Selector).


Reagent Selector
File Edit Tools Lists Help
Print New Save Filter Cluster Sort Data
Select
Shop
Collect
Reagent Core
New List 1

The Reagent Core defines the basic structural features that you want to include in your reagents. To create a Reagent Core, start ISIS/Draw, draw a Reagent Core, copy it, and then paste it into this box.

To select/deselect one reagent:	Click the reagent
To add selected reagents to a list:	Copy selected reagents, select list, Paste
To save selected reagents to a new list:	Click the Save button
To save all reagents to a list:	Click the Save button
To magnify a reagent:	Right-click and choose Zoom

Strategies for finding AMINES...
Strategies for finding ALKYL HALIDES...
Strategies for finding ALCOHOLS...
Strategies for finding other common reagents...
Hints on managing large lists of Reagents...
Using the List History to review and adjust your reagent selection...
Help Index...

Figure 3. "How do I?" menus (MDL's Reagent Selector).

OLUTION 1: IMPROVE THE USER INTERFACE

Since online help is merely a written apology for a poor user interface, let's try to improve the user interface and apologize less. At MDL, we use two approaches to achieve this solution, paper and electronic prototyping/testing, along with the design principles of John Bowie's Information-driven Design (IDD). The principles and methodologies of prototyping and usability testing are well known and widely available. We learned the principles of IDD initially from an article in STC's *Intercom* (Bowie 1996) and from subsequent workshops given by Bowie. The fruits of our labors will be seen in the results of the usability test, later.

SOLUTION 2: EMBEDDED HELP

Yes. This is good.

In addition to the examples of embedded help noted in "Fear and loathing," here are some more recent ones.

Text in the user interface (Adobe PhotoDeluxe)

PhotoDeluxe is an application that manipulates digital images. The application's audience is any member of the general public who is switching from film to digital images for their family and hobby images. With such a wide audience, this application needs to be usable. In the example, the user wants to remove "red eye" from an image. Initially, the user clicks a button labeled "Remove Red Eye." Then the user sees the information shown in Figure 1a. Here, text in the user interface confirms that, indeed, the user has found the correct function and then prompts the user with the "Start Here" label. When the user clicks the tab, the information shown in Figure 1b is displayed, and that information leads the user, step by step, through the procedure.

In the past, on MDL product teams, programmers resisted the notion of text in the user interface. They didn't know about the production paradox. This was unfortunate, because text in the user interface can guide and assist the user without ever needing to search through a help file.

Tooltips (MDL's Reagent Selector)

Users like tooltips (Spool 1996a). However, it is important to make the tooltips useful. In Figure 2, the tooltip for the somewhat vague icon, "Shop," indicates that the user can choose both chemical suppliers and prices for those reagents in the ACD databases.

"How do I?" menus (MDL's Reagent Selector)

Reagent Selector, in the Select mode shown in Figure 3, has three separate panels (the rightmost panel is cropped from the figure). Each panel has its own help menu, invoked when the user clicks the "?" icon in the panel of interest. The menu then provides information pertinent only to that

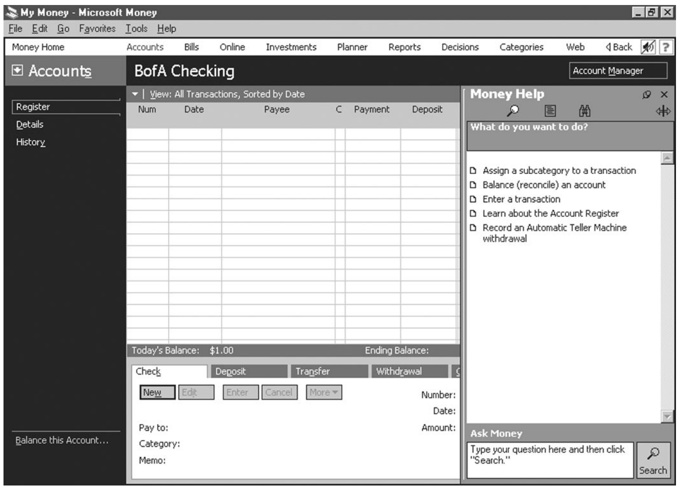

My Money - Microsoft Money
File Edit Go Favorites Tools Help
Money Home Accounts Bills Online Investments Planner Reports Decisions Categories Web Back
Accounts
BofA Checking
Account Manager
Register
Details
History
View: All Transactions, Sorted by Date
Num Date Payee C Payment Deposit
Money Help
What do you want to do?
Assign a subcategory to a transaction
Balance (reconcile) an account
Enter a transaction
Learn about the Account Register
Record an Automatic Teller Machine withdrawal
Today's Balance: $1.00
Ending Balance:
Check Deposit Transfer Withdrawal
New Edit Enter Cancel More
Number:
Date:
Amount:
Pay to:
Category:
Memo:
Balance this Account...
Ask Money
Type your question here and then click "Search."
Search

Figure 4. Embedded help panes (MS Money).

pane. These menus are similar to the "How do I?" menus used by Intuit in several of their products. A presenter from Intuit at the 1999 WinWriters Online Help Conference (Robertson 1999) noted that Intuit has had good test results with these menus. These are, indeed, help menus, but the important distinction is that they are context specific.

Embedded help panes (MS Money)

Figure 4 shows the embedded help pane in the 1999 version of Microsoft Money. The embedded help in this version was only partially context specific (that is, the help pane did not always change according to the actions of the user) because the developers chose to put more effort into developing natural-language search queries (which can be typed into the text box at the bottom of the help pane).

Drill-down links in browser-based applications

Figure 5 shows a tiny sliver of a browser-based application. Here, the user must choose a search type from the drop-down menu. Because the menu lists a number of unexplained, jargon-laced search types, the user would have difficulty making an informed choice (compared with a wild guess). However, by clicking the "Search type" hyperlink immediately adjacent to the drop-down menu, users can find all the information they need, progressively disclosed, on the search types available.

First, users find a one-paragraph definition. If more information is required, they can drill down to how-to instructions and detailed searching examples for the search type in question. As Nielsen notes in his article, "Inverted pyramids in cyberspace" (1996), this approach gives reluctant readers a good shot at getting the information they need very quickly and succinctly, with the ability to drill down for more detail, as needed.

Truly embedded

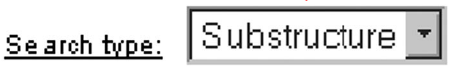


Figure 5. Drill-down links in browser-based applications.

Embedded?

There are also many other types of embedded help, including:

- ◆ Wizards (for a good example, see Intuit's Financial Planner)
- ◆ Audio cues (Intuit's Quicken)
- ◆ Video (Intuit's Financial Planner and TurboTax)

These various kinds of embedded help *seem* to be good solutions for providing end users with needed assistance in that they sidestep the user's reluctance to go to the Help menu. But *are* they good solutions?

At one time, manuals seemed a good solution. Then WinHelp invoked from the Help menu seemed a good solution. We've been here before! This article suggests that there are two crucial questions that we need to ask about embedded help (questions that we forgot to ask about manuals and WinHelp):

- ◆ Do end users actually *use* these forms of help?
- ◆ Do they actually find them *helpful* in doing their tasks?

OUR HYPOTHESIS

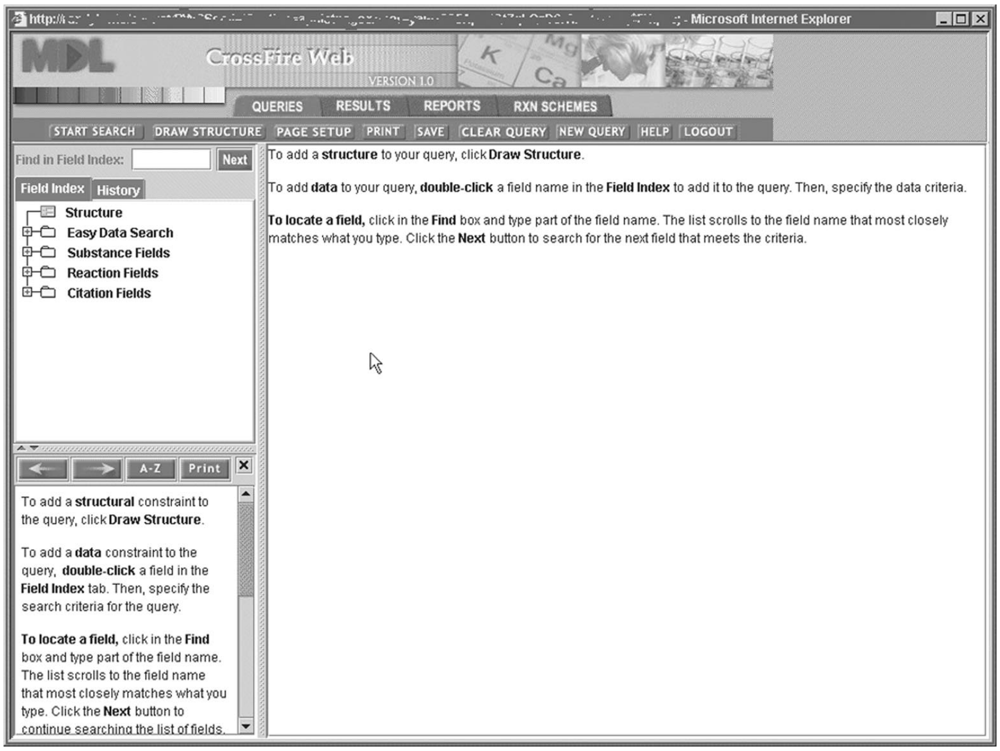
As noted in my "Fear and loathing" article, we hypothesized that end users will use, and find helpful, any form of help that incorporates the following five key characteristics:

- ◆ **Context-specific** Contains *only* the information relevant to the specific context.
- ◆ **Useful** Contains *all* the information relevant to the specific context.
- ◆ **Obvious to Invoke** The mechanism for invoking help must be obvious to the user.
- ◆ **Non-Intrusive** Invoked only when the user requests it. Does not distract the user's attention from their work before being invoked.
- ◆ **Easily Available** Help is just one or two clicks away. Does *not* require rummaging through help topics or using an index.

Different types of embedded help, as shown in the examples above, if designed carefully, can exhibit all five of the characteristics.

The opportunity to test this hypothesis came during the design and development of two associated browser-based applications:

- ◆ CrossFire Web, a browser-based database-searching application for searching over the Beilstein database,



To add a **structure** to your query, click **Draw Structure**.

To add **data** to your query, **double-click** a field name in the **Field Index** to add it to the query. Then, specify the data criteria.

To locate a field, click in the **Find** box and type part of the field name. The list scrolls to the field name that most closely matches what you type. Click the **Next** button to search for the next field that meets the criteria.

To add a **structural** constraint to the query, click **Draw Structure**.

To add a **data** constraint to the query, **double-click** a field in the **Field Index** tab. Then, specify the search criteria for the query.

To locate a field, click in the **Find** box and type part of the field name. The list scrolls to the field name that most closely matches what you type. Click the **Next** button to continue searching the list of fields.

Figure 6. Query page of CrossFire Web.

a database of over 8 million chemical structures and their associated data

- Draw for the Web, a browser-based chemical-drawing editor used for drawing search queries to insert into CrossFire Web

Working with the entire product team, we made extensive use of the principles of Information-driven Design to produce the user interfaces for both applications. During the design phase, we regularly measured usability by having MDL staff and then actual customers walk through typical user scenarios with paper prototypes.

In his comments on the paradox of the active user (Nielsen 1998), Nielsen states,

> *The "paradox of the active user" is a paradox because users would save time in the long term by taking some initial time to optimize the system and learn more about it. But that's not how people behave in the real world, so* ***we cannot allow engineers to build products for an idealized rational user*** *. . .* ***we must design for the way users actually behave.***

In our attempt to deal with the way users actually behave, we decided in our two new applications to

1. Make the embedded help an integral part of the user interface and, like any other part of the interface, open by default.

2. Make it bear as little resemblance as possible to WinHelp.

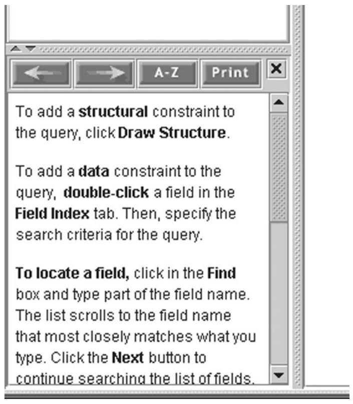
3. Ensure that the embedded help met the five key characteristics.

CROSSFIRE WEB

CrossFire Web is a browser-based front end for the Beilstein database. Chemists draw a structure, add data constraints, such as "melting point less that 90 degrees centigrade," and conduct a search. They then view the results.

The query page

Figure 6 shows the overall design. At the top is the banner, below which appears the now-ubiquitous tab bar. Each tab has several buttons associated with it, the buttons appearing on the button bar immediately below the tabs. There are no other menus. The rest of the screen is divided into three areas: A scrollable, expandable index of fields in the database in the top left corner; the embedded help window, immediately below the field index; and the work area, where the user builds up



To add a **structural** constraint to the query, click **Draw Structure**.

To add a **data** constraint to the query, **double-click** a field in the **Field Index** tab. Then, specify the search criteria for the query.

To locate a field, click in the **Find** box and type part of the field name. The list scrolls to the field name that most closely matches what you type. Click the **Next** button to continue searching the list of fields.

Figure 7. Embedded help topic in CrossFire Web.

a database query. (Initially, with no query created, the work area simply repeats the information in the embedded help.)

The embedded help pane

Figure 7 shows a close-up of the embedded help pane, with buttons for browsing, accessing the index (A–Z), printing, and closing the pane (X).

Let's examine this topic from the standpoint of the five key characteristics:

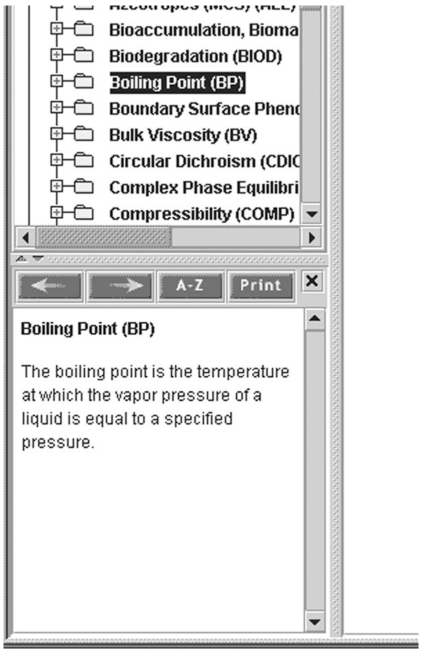
- **Context-specific** Yes. It describes the user's choices at this point; that is, how to build a query.
- **Useful** Yes, it shows all choices available at this point.
- **Obvious to invoke** Not an issue because it is already invoked by default
- **Non-intrusive** Would the user find this help pane, open by default, to be intrusive? Only the usability test will show.
- **Easily available** Yes. No searching is required to find the information the user needs at this point.

Context-specific help for database fields

Figure 8 shows the embedded help that is available after the user has single-clicked the "boiling point" field.

Again, let's consider this help in terms of the five key characteristics:

- **Context-specific** Yes. The user highlighted and clicked the "boiling-point" field and received information on this topic.
- **Useful** Yes. All the information that is needed at this point is provided; that is, the user is able to make the correct, educated choice from the information available and continue with the task at hand.
- **Obvious to invoke** It seems obvious, but only usability-test observations can confirm this supposition.



Boiling Point (BP)

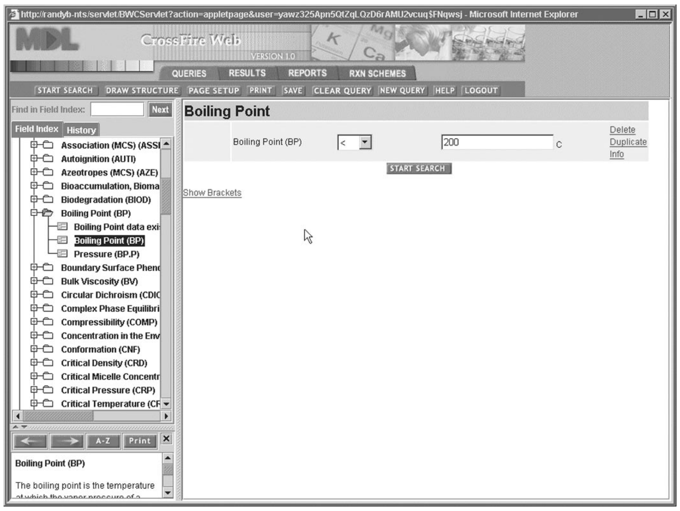
The boiling point is the temperature at which the vapor pressure of a liquid is equal to a specified pressure.

Figure 8. Embedded help for a database field.

- **Non-intrusive** Unknown. Again, the usability test will provide data to validate this supposition.
- **Easily available** Yes. The field information is provided without any searching required.

Adding a field to the search query

Figure 9 shows the result of double-clicking the "boiling-point" field. The field is now added to the work area on the right to form part of the search query. The user then types in the appropriate field values.



Boiling Point (BP) < 200 C

Boiling Point (BP)

The boiling point is the temperature

Figure 9. Adding a field to the search query.

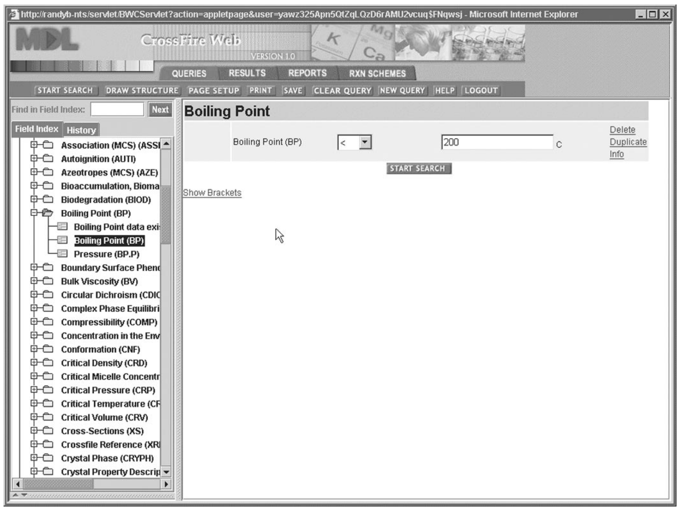


Figure 10. Closing the help pane.

The help topics are written using standard HTML. Thus, for any field, hyperlinks can be added to link to additional information, such as other HTML pages, PDF files, video, audio, show-me animations, and so on.

Closing the help pane

Figure 10 shows the result of clicking "X" in the embedded help pane to close it. After the help has been closed, would users be able to figure out how to open it again—that is, is the help obvious to invoke? We thought so (there is a Help button on the button bar), but only usability testing can confirm this assumption.

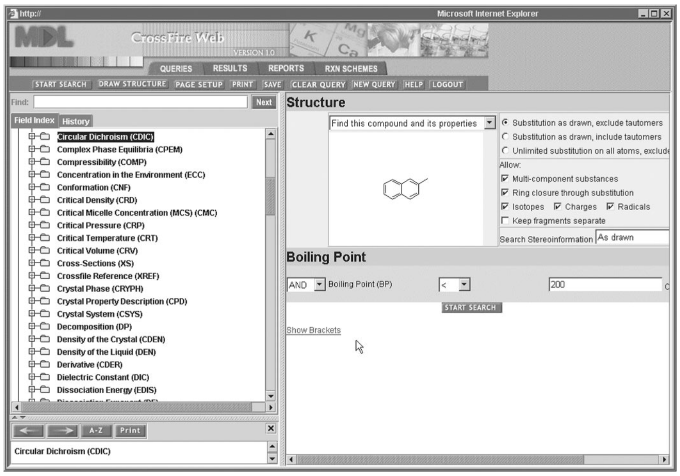


Figure 11. Resizing the help.

Resizing the help

Figure 11 shows that users can resize help by dragging the vertical bar. But is this feature obvious to invoke? Again, only usability testing can confirm this supposition.

The index

Not all information is context specific: There is often an amount of reference material (such as database conventions, and so on) that the user may want to access. As shown in Figure 12, we added an index to ensure that the help would be useful—that is, provide all the information relevant to any particular context.

Given the large amount of context-specific information and the form-filling nature of the application, we didn't see much need for an index. But only usability testing can confirm this hypothesis.

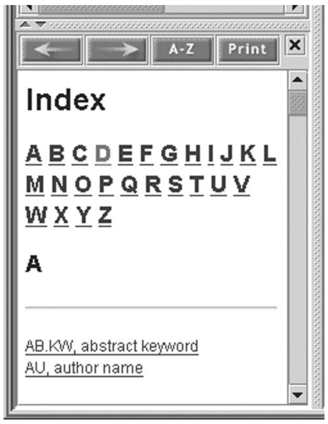
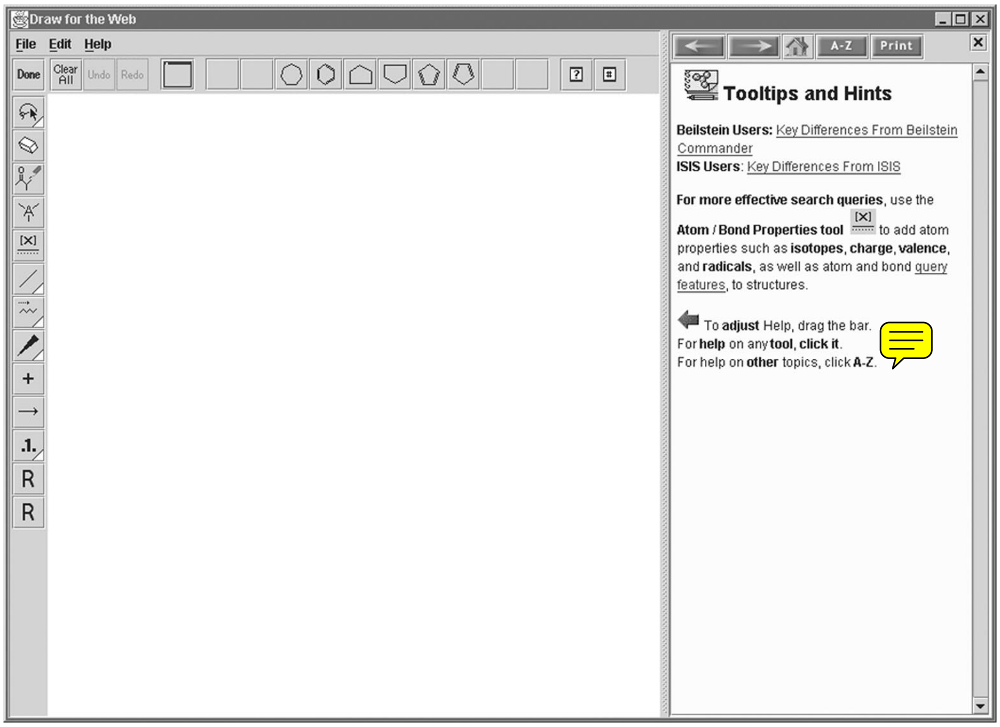


Figure 12. The index.



Tooltips and Hints

Beilstein Users: Key Differences From Beilstein Commander
ISIS Users: Key Differences From ISIS

For more effective search queries, use the **Atom / Bond Properties tool** [X] to add atom properties such as **isotopes, charge, valence,** and **radicals**, as well as atom and bond query features, to structures.

To **adjust** Help, drag the bar.
For **help** on any **tool, click it**.
For help on **other** topics, click **A-Z**.

Figure 13. Chemical-drawing editor with blank work area.

DRAW FOR THE WEB

In this chemical-drawing editor, users draw a chemical structure, using standard drawing conventions, which then forms part of the search query in CrossFire Web. In Figure 13, we see the default editor with a blank work area.

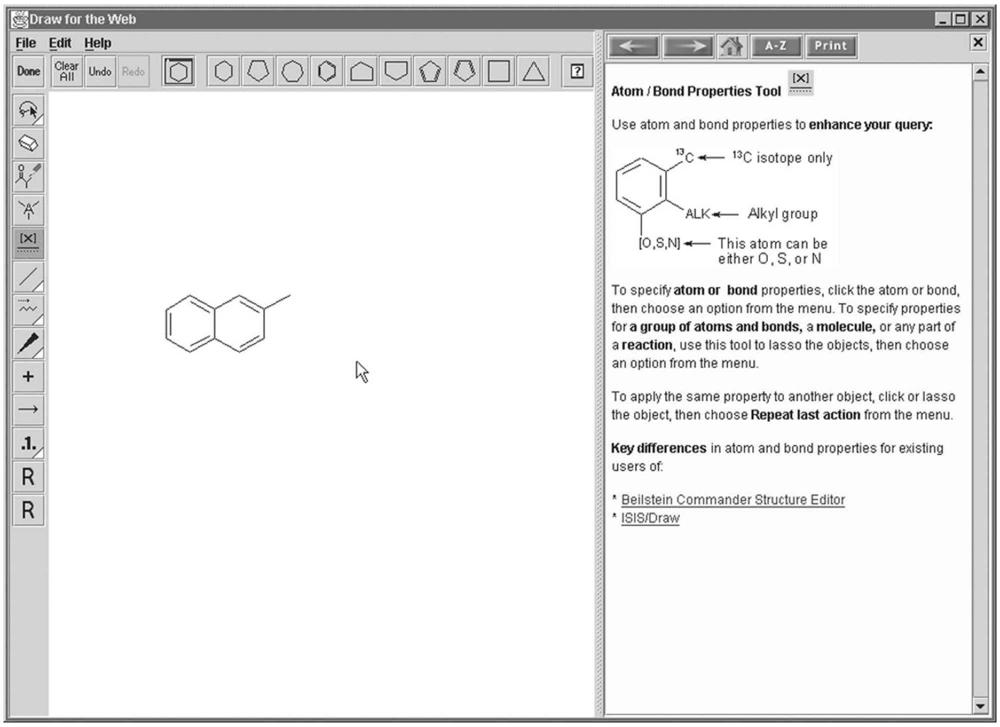
This time, the embedded help pane is on the right side. (In CrossFire Web, the help pane is on the left, which seemed to create a "natural" left-to-right workflow: Choose a database field from the left, and build a query on the right. In Draw for the Web, because most application toolbars appear on the left, it seemed "natural" to put the embedded help on the right. Although this placement is inconsistent between the two applications, no testers mentioned this inconsistency in the subsequent usability test.)

We titled the help pane "Tooltips and Hints" because users appear to prefer those names to "Help" (Spool 1996b).

In this opening, default topic, we wanted to draw attention to three things:

- ◆ In an attempt to combat the assimilation paradox, we wanted users of the earlier generations of our software to note some important differences.
- ◆ Based on earlier usability tests, we knew that users were likely (because of the production paradox) to create poor search queries. Thus, we tried to draw their attention to the Atom/Bond Properties tool and its associated query features.
- ◆ We wanted them to know a few important facts about using the help.

We decided to make this help open by default, for reasons similar to those for opening the help for CrossFire Web by



Atom / Bond Properties Tool

Use atom and bond properties to **enhance your query:**

^{13}C ← ^{13}C isotope only

ALK ← Alkyl group

[O,S,N] ← This atom can be either O, S, or N

To specify **atom or bond** properties, click the atom or bond, then choose an option from the menu. To specify properties for **a group of atoms and bonds,** a **molecule,** or any part of a **reaction**, use this tool to lasso the objects, then choose an option from the menu.

To apply the same property to another object, click or lasso the object, then choose **Repeat last action** from the menu.

Key differences in atom and bond properties for existing users of:

* Beilstein Commander Structure Editor
* ISIS/Draw

Figure 14. Embedded help as a tooltip for the Atom/Bond Properties tool.

Good--layering

default. Would it then, in the users' eyes, be non-intrusive? The usability test would reveal this.

Embedded help as "tooltip"

In Figure 14, the user has clicked on a tool, the Atom/Bond Properties tool (shown depressed on the left-hand toolbar). The embedded help has changed context and now acts as a "heavy-duty tooltip"—that is, the user can drill down to any amount of information of any type provided by the help authors (HTML pages, PDF files, video, animation, and so on) necessary to provide all and only the information pertinent to the current context.

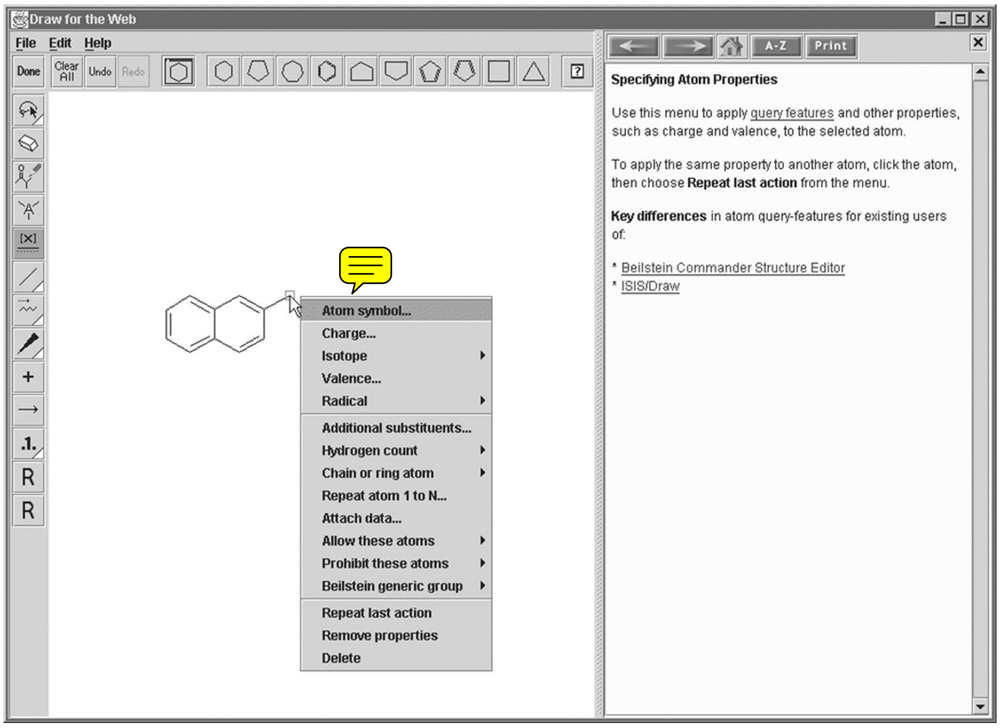
This help should also meet the five key characteristics:

◆ **Context-specific** Yes. The help provides information on the chosen tool only.

◆ **Useful** Yes. Through hyperlinks, users can drill down to all the information available on the use of this tool to help them make the correct, educated choice and continue with the task at hand.

◆ **Obvious to invoke** Uncertain. The default screen provided information on invoking help for tools. But will users read it? The usability test will answer this question.

◆ **Non-intrusive** Perhaps not. The help covers a significant part of the work area. We looked forward to seeing the usability test results on this item.



Specifying Atom Properties

Use this menu to apply query features and other properties, such as charge and valence, to the selected atom.

To apply the same property to another atom, click the atom, then choose **Repeat last action** from the menu.

Key differences in atom query-features for existing users of:

* Beilstein Commander Structure Editor
* ISIS/Draw

Figure 15. Context-specific help for editing atoms.

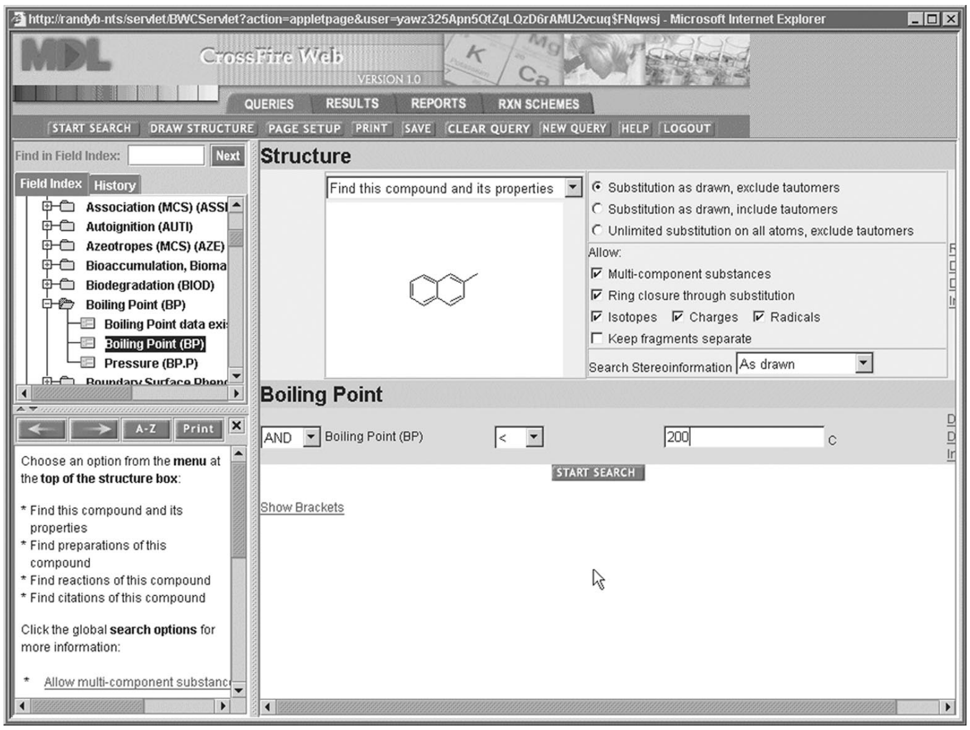
- **Easily available** Yes. There is no searching required to find information about the tool.

The user clicks on an *atom* to edit it in some way. A new menu appears in the work area (Figure 15) with all the available choices of functionality available for that tool/context. Note that the context of the embedded help has now narrowed to present information relevant to editing *atoms* only.

We believed that this help met the five key characteristics:

- **Context-specific** Yes. The help now provides information on editing atoms only.
- **Useful** Yes. By selecting hyperlinks, the user can drill down to all the information available concerning the use of this tool when editing atoms.
- **Obvious to invoke** Yes, if the help is already open. But what if users have closed the help? Will they be able to open it again (by clicking the "?" icon on the top toolbar). Only the usability test will answer this question.
- **Non-intrusive** Perhaps not. Again, the help covers a significant part of the work area. We would have to await results of the usability test to know whether this help met this key characteristic.
- **Easily available** Yes. There is no searching required to find information.

Having created a structure query, the user clicks "Done" to take the structure query back into CrossFire Web's Query



Find this compound and its properties

(•) Substitution as drawn, exclude tautomers
( ) Substitution as drawn, include tautomers
( ) Unlimited substitution on all atoms, exclude tautomers

Allow:
[✓] Multi-component substances
[✓] Ring closure through substitution
[✓] Isotopes [✓] Charges [✓] Radicals
[ ] Keep fragments separate

Search Stereoinformation: As drawn

AND Boiling Point (BP) < 200 C

Choose an option from the **menu** at the **top of the structure box**:

* Find this compound and its properties
* Find preparations of this compound
* Find reactions of this compound
* Find citations of this compound

Click the global **search options** for more information:

* Allow multi-component substanc

Figure 16. Query page in CrossFire Web, with structure query and boiling-point data constraint in place in the work area.

page, as shown in Figure 16.

In Figure 16, because certain structure-related search options are now available using the radio buttons and check boxes to the right of the structure, the embedded help has changed to match the current context and now explains the structure options.

The user can now click on the "Start Search" button to conduct the search and then view the results in the Results tab.

In summary, we believed that the two types of embedded help did comply with all five key characteristics; however, as noted above, there were some places where we were not entirely sure, because of the subjectivity of, for example, non-intrusiveness. The usability test would resolve these issues.

HELP-DELIVERY TECHNOLOGY

CrossFire Web

Before discussing the usability test, here is a brief description of the technology involved in developing the embedded help shown in CrossFire Web:

- ◆ Each help pane is one HTML file.
- ◆ We used Microsoft FrontPage Editor.
- ◆ The default help pane is hard-coded into the software.
- ◆ For the approximately 250 database-field help panes:
  - ◆ Each help pane has a filename the same as the database field name (for example, "boilingpoint.html").
  - ◆ The application looks in a "helps" directory for a filename the same as the field name.

Draw for the Web

Here is a brief description of the technology involved in developing the embedded help shown in Draw for the Web:

- ◆ Each help pane is one HTML file.
- ◆ We used Microsoft FrontPage Editor.
- ◆ On startup, the application reads an XML file, where it finds the name of the default help pane.
- ◆ For the tools, an XML file contains the name of the tool, filename of the icon, and the name of the help pane. For example:
  item name="AllPurpose DrawingTool" image="IconImages/Pencil.gif" dochelp = "AP_DEFAULT.html"
- ◆ To display the HTML help files within the application without opening a new instance of the Web browser, we used a third-party software package, CalPane, a sub-component of a Java development package called Calpa.

THE USABILITY TEST

We have formulated our hypothesis. We built embedded help incorporating the five key characteristics. Did it work? Or did we strike out for the third time?

The usability test was designed on the same lines as the earlier one described in "Fear and loathing" (Grayling 1998). For details, please see that article. To summarize, we:

- ◆ Tested 10 customers who matched the user profile.
- ◆ Positioned the test as "a usability test of our new *applications*" (no attention was drawn to the embedded help).
- ◆ Recruited a mixture of novices and self-reported "experts" (those with some experience of other MDL software). "Experts" would know some general concepts but would not know how to use the new software.
- ◆ Tested one user at a time.

TABLE 1: RESULTS OF THE USABILITY TEST

Tester #	Prior Experience?	Scenarios Attempted	1	2	3	4	5	6	7	Prompts by Facilitator
4	Y	7	√	√	√	√	√	√	√	
6	Y	7	√	√	√	√	√	√	√	
8	Y	7	√	√	√	√	√	√	√	
9	N	7	√	√	√	√	√	√	√	
1	Y	7	√	√	√	√	√	√	R	1 (S-3)
10	Y	7	√	√	X	√	√	√	R	
7	Y	6	√	√	√	√	√	R		1 (S-4)
2	Y	7	√	√	N	√	√	√	R	
3	N	7	√	√	N	√	√	√	R	1 (S-3)
5	N	6	√	N	N	√	√	√		

◆ *√ = Successfully completed the entire scenario.*
◆ *R = Ran out of time and didn't complete the scenario.*
◆ *X = Declined to attempt the scenario.*
◆ *N = Attempted but did not complete the scenario.*

◆ Gave them 7 real tasks, and approximately 2 hours in which to accomplish them.
◆ Asked them to verbalize ("talk aloud"), and we noted all comments.
◆ Did not answer any questions.
◆ Would assist with dealing with software bugs, misunderstanding scenarios, and with hints if the users became really, really stuck and were starting to get upset.
◆ Positioned an observer to the side of each tester to enable us to see where on the large 21″ monitor the tester was looking.

As designers of both the usability test and the embedded help itself, we were aware of the potential conflict involved. To mitigate this problem, we developed criteria that we thought could be unambiguously measured, we did not draw attention to the help (see the second bullet above), and we determined to submit the findings to a peer-review to confirm that other researchers examining the data would draw the same conclusions.

Typical scenarios

We determined a set of typical user scenarios, based on information accumulated over the years from site visits, survey results, customer contacts at trade shows, "Product Requirement" planning documents, MDL employees (many of whom come from industry and have the requisite domain knowledge), input from Marketing, and general comments from the field. In addition, discussions between Development and Marketing concerning new functionality (or changes to existing functionality) for our applications involve discussing appropriate user scenarios to put the functionality under discussion within context.

The scenarios were mostly of medium difficulty, representing very typical tasks that the user would want to perform with this software, as in the following example.

Scenario 1: Perform a search for all compounds with a boiling point less than 120 degrees (at a pressure of 760 Torr) for which bulk-viscosity data is available.

One scenario, however, was a complex task involving "retrosynthetic pathways," which are only of concern to certain types of chemists:

*Scenario 3: Find the synthetic path of [illustration of a compound]. Search for this compound by its Beilstein Registry Number, which is 5577185. If there is more than one possible synthetic route, I want to choose the synthetic route with the **fewest** number of steps and the path needs to be traced back until **commercially available** substances are available.*

RESULTS OF THE TEST

As noted above, MDL's Technical Communications User team and the CrossFire Web product team devoted much time and effort into building usability into the product from the very beginning of the design phase. As part of this process, we had conducted extensive in-house testing and had also brought customers in to test the paper prototype. Thus, we anticipated good test results.

We weren't disappointed. Table 1 shows the results of the test for the 10 subjects attempting 7 user scenarios. The

TABLE 2: USE OF THE EMBEDDED HELP IN CROSSFIRE WEB

Tester #	Prior Experience?	Scenario 2	Scenario 3	Scenario 4	Scenario 5	Useful?	Closed Help?	Used Index?	Comments?
4	Y		1		1	Y	N		—
6	Y		1			Y	N		—
8	Y		1		1	N	N	Y	—
9	N		2			Y	N		—
1	Y		2			Y	N		—
10	Y			2	1	Y	N		—
7	Y	1	1			Y	N		—
2	Y					—	N		—
3	N		4	1	1	N	N	Y	—
5	N		1			N	N		—

◆ *Numbers indicate the number of times the tester used the embedded help.*

testers were numbered from 1 to 10, in the order in which we scheduled them for testing. The testers have been reordered in the table to group the results into similar patterns of success.

Comments on the test results

The first "group" (Testers 4, 6, 8, 9, 1, 10, 7) all did very well, considering that they had never seen this application before and were also working under the pressure of public scrutiny.

- ◆ The first four testers shown in the table completed all the scenarios successfully with no prompting from the facilitator. Tester 9 completed all the scenarios successfully, despite having no previous experience with any of our software.
- ◆ Tester 1 got slowed down by Scenario 3, needed one prompt, and then completed it. However, this caused this tester to run out of time in Scenario 7.
- ◆ Tester 10 declined to try Scenario 3. This tester did not exactly fit the user profile and had no experience with tracing retrosynthetic pathways. (During the recruiting phase, we had contacted customer sites, explained our needs to our contact person, and received from the contact the names of suitable candidates. However, we did not reconfirm with the candidates themselves that they completely fit the user profile we were looking for. We later realized that this was an error on our part.)
- ◆ Tester 7 slowed down on Scenario 4 (the only tester to do so) and needed one prompt to continue on to success. This tester ran out of time in Scenario 6 and thus did not complete the last two scenarios.

The remaining three testers (2, 3, 5) were not as successful.

- ◆ Tester 2 was a self-identified expert with other MDL software and had many preconceived notions as to how the software "should" work. The tester became irritated when the software didn't behave as expected. This was a vivid and graphic demonstration of Carroll and Rosson's assimilation paradox made flesh: The tester's prior knowledge was hindering use of the new software. The tester gradually got more irritated and eventually gave up on Scenario 3, despite experience with retrosynthetic pathways. Given the time lost on this scenario, the tester subsequently ran out of time in Scenario 7 but was well on the way to completing it.
- ◆ Tester 3 was not familiar with retrosynthetic pathways: another case where the tester did not exactly fit the user profile. The tester completed about half of the scenario but did not understand what then needed to be done; so the facilitator suggested that the tester move on. Given the

slowdown in Scenario 3, the tester subsequently ran out of time in Scenario 7.

- Tester 5, who had never used any of our software, had difficulty using Draw for the Web in Scenario 2 (which caused us to subsequently recommend a small change in the user interface). Tester 5 was also not familiar with retrosynthetic pathways (the third user who did not completely fit the user profile), so this user began to thrash about in Scenario 3. The facilitator suggested that the tester move on to the next scenario. The tester should have been prompted by the facilitator in both Scenarios 2 and 3; however, the facilitator failed to do this for reasons unknown.

Evaluating the test results

Overall, we considered the test very successful. Testers were seeing this interface for the very first time and were operating under some pressure. Despite these facts, 7 of the 10 testers completed all but 3 of the 47 scenarios they attempted, requiring only two prompts from the facilitator to do so. They were also well on the way to completing 3 more scenarios before they ran out of time. The remaining three testers all failed to complete Scenario 3; however, two of them did not fit the user profile. Even this group successfully completed 14 of the 20 scenarios attempted, needing only one prompt from the facilitator to do so.

What part did the embedded help play in the success of the test? It is impossible to separate the effects of the extensive user-interface design work from the embedded help. Indeed, we always viewed the embedded help as an integral part of the interface design. We determined that useful criteria for evaluating the success of the embedded help would be

1. Did the testers actually use the embedded help at any time?

2. If they used it, did it provide the information they needed for the context?

USE OF CROSSFIRE WEB EMBEDDED HELP

Table 2 shows how the testers used the embedded help in CrossFire Web. (The testers are shown in the same order as in Table 1.)

Numbers indicate the number of times the tester used the embedded help.

Did the testers use it?

As Table 2 shows, users did use the embedded help, mostly in Scenario 3, a result that might be expected because that was the most difficult scenario. Only one of the 10 testers, Tester 2, did not use the CrossFire Web embedded help. This tester certainly needed it, judging by the test results in Table 1. We had already seen this user ensnared in the assimilation paradox. Was the tester also ensnared by the production paradox?

Tester 5 used the help, but only once. Again, given the test results, the tester probably should have used it more often.

Was the help useful?

Tester 8 did not find the help useful on one of the two occasions they used it. The user was looking for information on the use of wildcards in searching, information that was missing from the help. (This omission was subsequently fixed.)

Was the help useful to Testers 3 and 5? No, it wasn't because the help text was written to a particular audience—one that understood and used the concepts of retrosynthetic pathways—and these testers did not fit the user profile for Scenario 3.

Was the help non-intrusive?

Many product-team members were initially doubtful that this kind of embedded help would be non-intrusive. Our measure for this characteristic relied on whether the testers closed the help or made any negative comments about it.

As Table 2 shows, no one closed the help, not even Tester 2 who declined to use it. For Scenario 3 (retrosynthetic pathway), most testers needed more room in the workspace to see the material; so they reduced the size of the help pane by dragging it. However, no one actually closed the help or attempted to do so.

Similarly, no testers made any negative (or positive) comments about the embedded help. *For us, this was most valuable information*. One of the most common objections made by product-team members to embedded help of this type is that they thought it would be intrusive. For these 10 testers, at least, that was clearly not their perception.

Was the help obvious to invoke?

We were not able to identify whether the help, once closed, would be obvious to reopen because none of the testers closed it.

Was the help context specific?

As demonstrated in earlier sections, the help was always specific to the context.

Was the help easily available?

Did the help require rummaging through help topics or using an index? The index was used by 2 testers for a total of 3 times in 67 scenarios attempted. On all 3 occasions, the

TABLE 3: USE OF EMBEDDED HELP IN DRAW FOR THE WEB

Tester #	Prior Experience?	Scenario 1	Scenario 2	Scenario 7	Useful?	Closed Help?	Used Index?	Comments?
4	Y	2	3		Y	N		—
6	Y		5		Y	N		—
8	Y		2	1	Y	N	Y	—
9	N	1	1		Y	N		—
1	Y	1	1		Y	Y	Y	—
10	Y	1			Y	N	Y	—
7	Y	1	1		Y	N	Y	—
2	Y	2	1		Y	N	Y	Y
3	N	3	4		Y	N	Y	Y
5	N	3	3		N	Y		—

testers were looking for definitions of terms, which were easy to find. Given this small number of uses of the index, we believe that the help substantially complied with this characteristic.

USE OF DRAW FOR THE WEB HELP

Table 3 shows how the testers used the embedded help in Draw for the Web. (The testers are shown in the same order as in the previous tables.)

Did the testers use it?

All the testers used the embedded help in Draw for the Web. They mostly used it in Scenarios 1 and 2, a fact that is to be expected as part of the learning curve for this application. Even Tester 2, who ignored the embedded help in CrossFire Web, used the embedded help three times in Draw for the Web.

Tester 5—who stated at the luncheon after the test, "I don't read helps"—used the help no less than six times! It may well be that this tester didn't perceive the embedded help in the same light as WinHelp, a fact that could be interpreted as a major success for our embedded help design. Alternatively, this may simply be an example of the phenomenon noted in the literature (Spool 1997; Nielsen 2001b) that users often don't *do* what they *say* they do. This phenomenon highlights one of the major advantages of rigorous usability testing over surveys. It is doubly important for technical communicators because the Marketing departments of software companies often use surveys, along with e-mail and face-to-face contact, to elicit information on how customers use help or documentation. They then expect the documentation department to act on this information, self-reporting that may well be entirely inaccurate.

At first glance, it might seem strange that no testers used the help in Scenario 3, the scenario that caused the most difficulty. However, most of the work involved in this scenario took place in CrossFire Web; the work required in Draw for the Web for this scenario was relatively trivial and, by the time the testers had competed the first two scenarios, well understood.

Was the help useful?

Only one tester, Tester 5, did not find the Draw for the Web help useful. This tester had trouble drawing chemical structures as search queries. The fact that English was a second language for this tester may also have been a factor. For the other 9 testers, the help did assist them effectively.

Was the help non-intrusive?

As with CrossFire Web, our measure for this characteristic relied on whether the testers closed the help or made any negative comments about it.

- ◆ Tester 1 closed the Draw for the Web help immediately on entering the application but then opened it on the two occasions that help was needed. This was the only tester to follow this pattern.

- Tester 5 closed the help immediately on seeing it but then opened it soon after and kept it open for the remainder of the test.

Only one tester made a negative comment about the help for Draw for the Web: Tester 2 stated during the test that "help should not be open by default." This was the only tester to say so. However, at no time did this tester close the help.

The only other comment was a positive one: Tester 3 stated at lunch, "I like what I see. I just need to overcome the learning curve. I liked that the help was always there and was 'dynamic.'" Again, this is important ammunition when dealing with product-team objections.

Was the help obvious to invoke?

As previously noted, 2 of the 10 testers did close it. Both easily opened it again.

Was the help context specific?

As demonstrated earlier, the help was always specific to the context.

Was the help easily available?

In Draw for the Web, six of the 10 testers accessed the index for a total of 7 times. This would suggest that the help did not always meet the "easily available" criterion. However, 2 of the 7 occasions of accessing the index were to find information about an "Undo" function. In the early scenarios, testers made frequent drawing errors in Draw for the Web, and many of them searched the user interface for an undo command. However, at the time of the test, this function was not implemented and hence not visible. (It was added prior to the software release.) Thus, discounting the two searches for "Undo," the 10 testers accessed the Draw for the Web index only 5 times in all over the 67 scenarios attempted. Given this very low number of occurrences, we assume that the help generally complies with the "easily available" characteristic.

CONCLUSIONS

Our prior research, testing, and review of the literature confirm that *users don't like conventional help* (WinHelp invoked from the Help menu). This may be difficult news for practitioners highly skilled at producing WinHelp files to hear. It was certainly difficult news for us at the time. Watching testers in action is perhaps the only cure. It was clear from the expression on the face of a new staff member in our department who simply didn't believe us when we pointed out this awkward news that observing a usability test had demonstrated the validity of our conclusion.

The best solution is to stop creating elaborate written apologies for poor interfaces and begin to build usable ones instead. Some kind of embedded help will probably still be needed to provide information at all contexts to enable users to make informed choices. Our testing with two embedded help systems that incorporated the five key characteristics did meet our two criteria:

1. The testers did use this form of embedded help.
2. They found it useful, that is, it provided all the information they needed for that context.

Equally important, the well-documented resistance to conventional help (WinHelp invoked from the Help menu) was scarcely visible.

The test results also provide valuable data (ammunition) for technical communicators trying to overcome objections within their companies to using embedded help:

- *No* testers found the help intrusive, although both help systems were *open by default.*
- Eight of 10 testers *never closed the help* (although most resized it in Scenario 3 to make more room in the workspace).
- Only 2 of the 10 testers made any comment about the help (one of which was very positive).

As a profession, we badly need more published testing of the various types of embedded help so that we can all confidently move forward, sure in the knowledge that we are indeed fulfilling our mission of assisting our users to use our applications effectively and efficiently. We cannot fulfill this mission if our work lies, unseen and unused, in a WinHelp file. TC

ACKNOWLEDGMENTS

A subset of this information was originally presented at the WinWriters Online-Help Conference, Santa Clara, CA, March 2001.

Many thanks to the members of the CrossFire Web product team (Project Manager: Tonny Sasse, MDL Development) for embracing both the information-driven design techniques of user-interface design and our proposals for embedded help.

Many thanks also to the End-User TechCom team (Nicki Davis, David Krause, Frank Ripp, and Kimberly Steele) for their close-knit teamwork in designing and creating the embedded help and preparing for and conducting the usability test.

TRADEMARKS AND SERVICEMARKS

Intuit, Quicken, Financial Planner, are registered trademarks and/or registered servicemarks of Intuit, Inc. Microsoft is a registered trademark and FrontPage is a trademark of Microsoft Corporation.

REFERENCES

Bowie, J. 1996. "Information engineering: Using information to drive design." *Intercom* 43 (May):8.

Carroll, J. M., and M. B. Rosson. 1987. “The paradox of the active user.” In *Interfacing thought: Cognitive aspects of human-computer interaction*, ed. J. M. Carroll. Cambridge, MA: MIT Press/Bradford Books, pp. 80–111.

Grayling, T. 1998. “Fear and loathing of the help menu: A usability test of online help.” *Technical communication* 45, no. 2:168–179.

Hackos, JoAnn. 2000. “Our customers hate using help.” 8th Annual WinWriters Online Help Conference.

Hughes, Michael. 2000. “Beyond stepped procedures.” 8th Annual WinWriters Online Help Conference.

Krug, S. 2000. *Don’t make me think! A common sense approach to usability.* Indianapolis, IN: Macmillan USA.

Nielsen, Jakob. 1996. “Inverted pyramids in cyberspace.” http://www.useit.com/alertbox/9606.html

Nielsen, Jakob. 1998. “The paradox of the active user.” http://www.useit.com/alertbox/activeuserparadox.html

Nielsen, Jakob. 2001a. “Error-message guidelines.” http://www.useit.com/alertbox/20010624.html

Nielsen, Jakob. 2001b. “First rule of usability? Don’t listen to users.” http://www.useit.com/alertbox/20010805.html

Redish, Janice T. 1999. “Expanding our understanding of users: Lessons from minimalism and beyond.” 7th Annual WinWriters Online Help Conference.

Robertson, Jenny. 1999. “Developing onscreen user assistance.” 7th Annual WinWriters Online Help Conference.

Schriver, Karen. 1999. “Evolution of software user assistance from the perspective of information designers and users.” 7th Annual WinWriters Online Help Conference.

Spool, J. M. 1996a. “Effective tool tips.” *Eye for design* 3, no. 1:1.

Spool, J. M. 1996b. *“Tips and Hints.” Eye for design* 3, no. 3:5.

Spool, J. M. 1997. “Online help: Sometimes doesn’t help.” *Eye for design* 4, no. 3:9–10.

TREVOR GRAYLING is director of Technical Communications at MDL Information Systems. He has spearheaded the design of documentation sets, help files, and embedded help for various graphical and database applications. To date, his team has won five STC awards. He introduced usability testing to MDL in 1990. He served as a judge for the STC Northern California Online Help Competition for 3 years. He earned a BSc (Hons) in mathematics at Hull University and a graduate certificate in education at Exeter University, both in the UK. He has been a technical communicator for 16 years. Contact information: trevor@mdli.com