On Help Systems In General

Updated February 27, 2003

David E. Liske
Delmar Computing Services
Tipton, Michigan, USA

In the eras of Windows 3.x and earlier versions of Windows 95, the only help system people worked with or even knew about was WinHelp. Problems started with the transition to Windows 95, when developers and users alike had to learn to deal with WinHelp 4.0's separate dialog with the Contents, Index, and Find tabs. (For example, VB developers still have a difficult time calling the Contents tab from a program, and the Common Dialog control has yet to be fully compatible with the WinHelp tabbed dialog.) In 1997, Microsoft released HTML Help, which has become the standard for Windows 98 and 2000, and is what the MSDN system for Office 2000 and Visual Studio 6 are based on.

In the time since Windows 95 was released, quite a few help systems have appeared. RoboHELP allows for the creation of both WinHelp and HTML Help files. It also is capable of creating what's called WinHelp 2000, which is a WinHelp system that looks and acts like HTML Help. It also generates WebHelp, a browser-based cross-platform help system that looks like HTML Help but isn't context-sensitive. Doc2Help can generate many of the same types of files (except a WinHelp 2000 equivalent), and has a similar cross-platform solution called InterHelp. The HDK system has been around a long while, and makes an even more configurable WinHelp solution than WinHelp 2000. There's also the cross-platform JavaHelp (which both RoboHELP and ForeHelp can create) and the now-defunct-but-still-available NetHelp, both of which look and feel like HTML Help but have different technologies underneath. There are probably even more help systems out there for proprietary applications. And then there are the man pages used in Linux ... and UNIX ... and the Mac help system ...

A few questions come from all of this:

A. As a user, which help system do you prefer most or least, and why? And if given the choice, what would you like to see in a help system?

B. As a developer, which help system do you most prefer to develop with, and why? What problems have you seen in implementing various help systems from specific development languages? And if given the choice, what would you like to see in a help system?

For example, as a user, I least prefer the MSDN system as implemented in Office 2000 as it's considerably more clunky than it needs to be. But as a developer, I use MSDN from the three CD's as it gives me the information I need. I prefer to develop with HTML Help since I can be much more creative with it. However, some users don't like it as a matter of choice simply because it's dependent on IE being installed on the system (even though IE doesn't need to be the default browser), so I need to be careful. And when I'm developing controls for use with VB5, I need to use WinHelp for those users who don't have IE installed. If they indeed have IE installed, I can use Innovasys Document! X tool for generating WinHelp stubs that call HTML Help topics ...

As for what I'd like to see, that would be a fully cross-platform help system that isn't dependent on anything in particular one way or the other.

So, what about the need for a help system for any given application? Is it even necessary? Some managers will argue that since help isn't used by everyone who uses a given application, it's not cost-effective to develop help at all.

The single question to ask of such a manager is this: "Would you rather spend the money now for a decent help system to give the users what they need to know when they need it, or spend the money later for a larger support staff to handle the incoming requests for assistance for this application?"

One of the more common problems in this area is the gap between developers and help authors. Developers either don't believe help is necessary (apparently because they believe their applications to be so incredibly intuitive) or they write it themselves (which throws the professional help authors for a loop because they apparently don't feel the developer can be objective to their own work). Help authors feel

there should be a help system of some kind for every application written.

I'm a developer who's also a help author. I'll side with the help authors on this last point, while still writing my own help files.

At issue here more than anything is that some user somewhere is going to need help with your application. Rather than paying a support person by the hour to sit by the phone and wait for that user to call, spend the time to ship the support with the application in the form of a help system. (Don't believe for a moment that such a call can just go to the developer. Excellent support personnel require more people skills than most developers are trained to have.) If you're a developer and you can't be objective, hire someone to write the help for you. If you'll be writing your own help system, be excruciatingly brutal with your own work. When you have your application Alpha and Beta tested, have the testers also test the help system. And when the comments come in from these testers, throw your ego out the window and do what's necessary for the user.

Help is a required part of an application. Many times, developers and managers alike make it the last thing on the list of items to be done. To be done right, help should be developed concurrently with the application itself, not as a rushed afterthought. Make your help system as complete as you possibly can, and when you've run out of things to include, realize that you've probably missed something, and go over all of it again.

You'll definitely save money on the support staff in the long run.