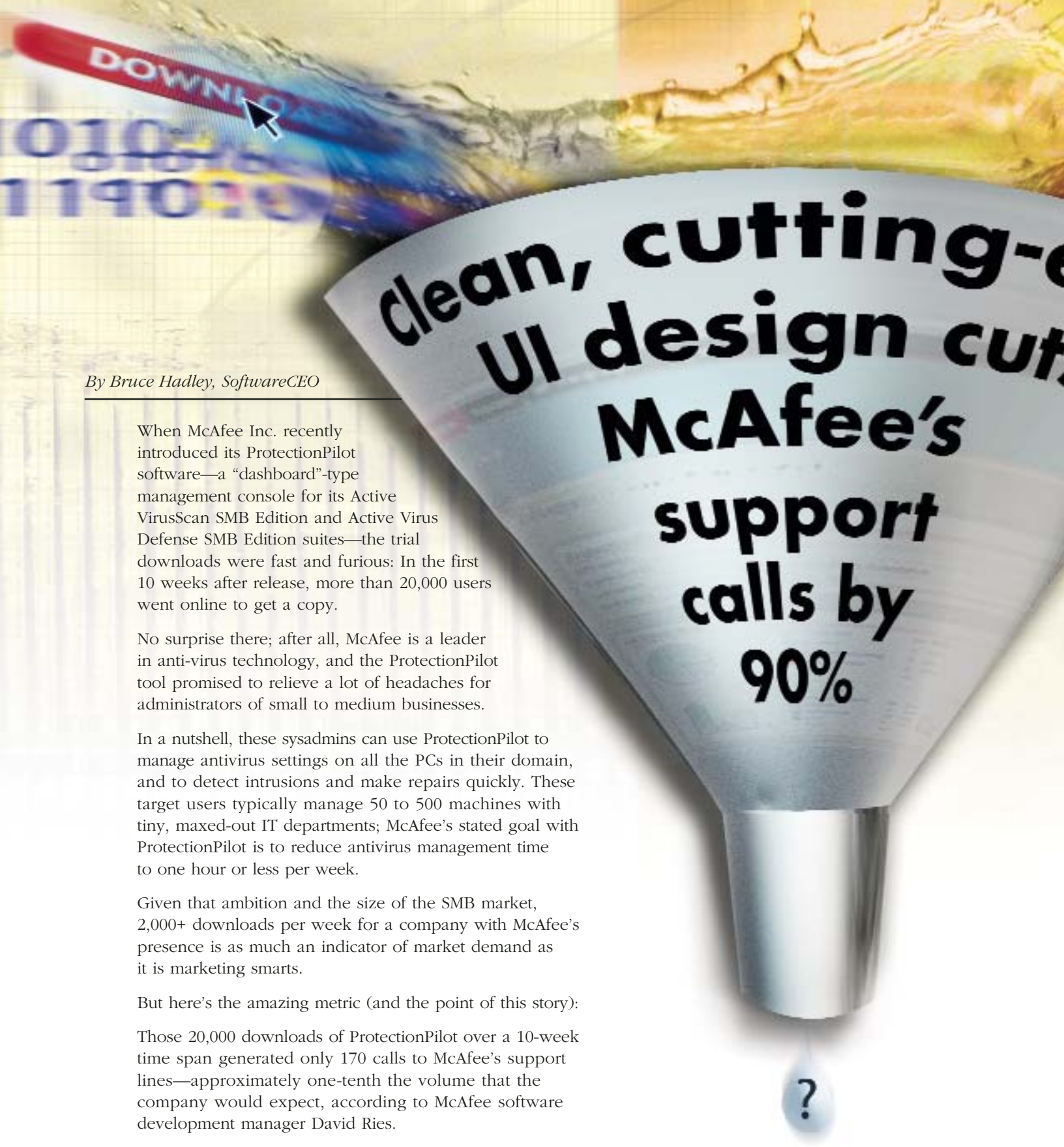


By Bruce Hadley, SoftwareCEO

When McAfee Inc. recently introduced its ProtectionPilot software—a "dashboard"-type management console for its Active VirusScan SMB Edition and Active Virus Defense SMB Edition suites—the trial downloads were fast and furious: In the first 10 weeks after release, more than 20,000 users went online to get a copy.

No surprise there; after all, McAfee is a leader in anti-virus technology, and the ProtectionPilot tool promised to relieve a lot of headaches for administrators of small to medium businesses.

In a nutshell, these sysadmins can use ProtectionPilot to manage antivirus settings on all the PCs in their domain, and to detect intrusions and make repairs quickly. These target users typically manage 50 to 500 machines with tiny, maxed-out IT departments; McAfee's stated goal with ProtectionPilot is to reduce antivirus management time to one hour or less per week.

Given that ambition and the size of the SMB market, 2,000+ downloads per week for a company with McAfee's presence is as much an indicator of market demand as it is marketing smarts.

But here's the amazing metric (and the point of this story):

Those 20,000 downloads of ProtectionPilot over a 10-week time span generated only 170 calls to McAfee's support lines—approximately one-tenth the volume that the company would expect, according to McAfee software development manager David Ries.

And, roughly a third of those support calls were actually pre-sales questions, Ries says, from people who wanted more information about what the product could do; they didn't need technical help.

OK, let's let this sink if for a moment:

You've just introduced version one-dot-oh of a brand-new software product. You put it up on your site for its production field trials—i.e., this is the first time it will be used by "unschooled" users—those who have no prior exposure to it.

If you're a normal software company with a normal product, you beef up your tech support resources: Your tech reps, SEs, QA people, and developers are all put on call to deal with the anticipated onslaught of calls from frustrated users who can't figure out how to use your software (as well as those who refuse to read manuals, FAQs, and help files).

As you watch the downloads spool off at the rate of 2,000 per week, you should expect—if your experience is anything like McAfee's with its enterprise products—to get about 100 support calls from each week's downloads.

Instead, you get just 10 support calls per week—just two per business day. You can send those extra tech reps home, and let the SEs get back to selling, the QA people back to testing, and the developers back to developing.

For most B2B software vendors, this would be a very happy outcome indeed. Consumer tools are accustomed to lighter support loads, but ISVs targeting business markets—especially in the rarefied world of network management—don't often get off this easy.

So, how'd McAfee do it? How did they maintain hot sales activity and yet reduce their initial support load by 90 percent?

Two words: user interface.

This was no lucky accident, of course; McAfee made user interface (UI) design a prime directive of ProtectionPilot very early on.

"We knew that the sweet spot [for ProtectionPilot] is the sysadmin who's managing 250 machines," Ries says. "These guys have a lot of different jobs, and they don't have a large staff; they generally have one IT guy who doesn't have a lot of time.

"At the outset, we had two big concerns: We designed ProtectionPilot so that the person using it would need to spend less than an hour a week maintaining his organization's antivirus protection.

"Second, because of the large number of users, we wanted to keep our support calls low. No matter how trivial, every support call has some costs associated with it. Installation should be easy, and initial use should be intuitive."

The bottom line, as far as Ries is concerned, is straightforward: Focusing on the design of the product had a significant impact on the cost of supporting the product.

Of course, there's a very positive sales benefit to this, too: ProtectionPilot's UI design has already generated favorable reviews in the U.S. and Europe, with ease of use a common theme.

"By focusing on the user's experience, we were able to deliver a favorable user experience," Ries says.

Well, that sounds easy, right?

Hardly. Here are 23 detailed tips we gleaned from McAfee and their external UI design team; if you can master this list, you've got a great chance at selling more software and spending less to support it. ➔

20,000 downloads of ProtectionPilot over a 10-week time span generated only 170 calls to McAfee's support lines—approximately one-tenth the volume that the company would expect... And, roughly a third of those support calls were actually pre-sales questions.

UI tip #1:

Start the UI design before you build the product. An early focus on user experience meant that McAfee included users at a very early stage in product development. This may sound obvious, but it's surprisingly difficult for most software developers; products are typically kept under wraps until there's a beta release.

Big mistake, Ries says; you need to get that user feedback before you build anything real. "Early in the development stage we developed a mockup or prototype of the product," he says. "It had some live links to make it look and feel like a real product.

"We came up with a set of essential tasks—the list of things that the product had to do—and then did the prototyping in-house and figured out the workflow."

But it wasn't real; it was a sketch, or, to use Ries' example, a storyboard. "Rather than shooting a whole movie and showing it to people to see if they like it, we put together a storyboard. That prototyping was very, very important. All the prototyping was done in HTML, which meant you could run it in a browser.

"We created a facade, like a movie-set western town. A lot of the things that users saw were not real: For example, you might see a picture of a report—we flung a jpeg of a log file out there—rather than an actual report.

"The point is to get something out there quickly. It's really hard to jettison a week's worth of development work; but if you've just whacked together a picture, your team is not emotionally attached to it."

UI tip #2:

Understand your software from a user's standpoint. Of course, you can't prototype anything—at least not well—until you understand what it is the user will be able to accomplish with your software.

"You have to have a very clear understanding of what the product is supposed to do, from a task perspective rather than features," Ries says. "You're really talking about workflow here.

"For example, the feature is that I can run a report. But at the end of the day, users don't care about running reports, they just want to be able to make a decision—and the report answers what that decision is supposed to be.

"For example, we dropped reporting as a feature in the small business product, and instead created an active dashboard. So yes, it generates reports, but not in the same way that most people think about reports."

When you're prototyping, the UI consideration is about user accomplishment, not product features, says McAfee engineering director Mark Wyman: "What is the end-user trying to accomplish by interacting with your product?"

UI tip #3:

Get feedback through task-oriented use. The whole point of UI design is usability. If that sounds like another "Well, duh," humor us for a moment.

Most developers, when (and if) they include users, look for feature feedback. You want to know what needs to be in the software, what can wait for the next release, and how you stack up against the competition.

Trouble is, that sort of quest does nothing to improve your software's usability. You need to stay focused, Ries says, on real and specific tasks that your users will need to perform.

"We sent the mockup to real IT administrators and asked them to play around with it," he says. A week later, we asked them to answer some questions—not about functionality of the product, but strictly about the tasks they had to perform.

"For example, we asked them if they could figure out, via the ProtectionPilot prototype, how many machines are up to date, and which ones were not up to date. Could they tell us how they'd perform an on-demand update?"

Another example: The McAfee team asked users how they knew when a new virus threat was out there in the world. The answer: Either it gets detected by their anti-virus software, or they saw a report on CNN, or they "heard it from a buddy."

That kind of feedback told the McAfee team that it would be desirable to deliver more data to them via ProtectionPilot—to make the software an answer to their information needs.

McAfee works hard to protect the good relationship it has with this elite group of users. "We are really careful to use their time wisely," says Pam Conrad, project manager for mile7, the Portland, Ore.-based design and research consultancy that worked with McAfee.

"Our calls with the users are generally weekly, and we send them information at least two days in advance. If we're trying to test something specific, we'll give them an agenda for the upcoming call, and backup information if needed."

In addition to the small group meetings, McAfee and mile7 also do one-on-one calls. "We'll often break out and do calls with individuals," Conrad says.

"If there's a particular scenario we want to test, we'll send them the scenario, then use MeetingPlace so that we can actually observe what they're doing as they go through the scenario, and talk to them as they're doing it. We try to keep these calls to about half an hour, but they often take longer."

UI tip #4:

Segment the process into logical chunks. The McAfee team created four very specific phases within its 14-week ProtectionPilot development; the goal was to map UI feedback from users very specifically to the product's development maturity.

"The way you plan the iterations is key," says McAfee engineering director Mark Wyman. "You want to get the UI nailed first, and then fill in behind it."

"In each iteration we dropped to our customers, we had a specific focus and specific questions for the user," says Ries, "which helped us determine whether we'd successfully passed each milestone.

"Over the span of the development, we gave them four drops of the product to try out in their environment. In a couple cases where we got negative feedback, we knew what we had to tweak."

McAfee's segmentation for ProtectionPilot went like this:

Phase 1: In the first phase, when users were looking at a mockup, the questions to be answered were more broad: Does it look like it is going to work? Do we have the scope of the product right?

"This was about the general scope of the product—the implied functionality that it provided," Ries says. "For example, you have a help button that shows you're going to have help, even if that only brings up a picture."

Phase 2: In phase two, McAfee wanted to measure users' first exposure to the product: Can you install it and get it running?

"At this point we had a real installer, and you could accomplish the core tasks, with live code," Ries says, "but some of the things we showed you were still pictures."

Phase 3: In phase three, the task-oriented questions drove toward product usability: Are there enough options and settings? Does it work for you?

"Here we allowed user customization and changes," Ries says. "You could modify properties, you could add real machines and tasks, but a lot of the things weren't operational like error messages and boundary conditions.

"At this point, we were still a little fuzzy on performance timing issues; we were still tweaking the product."

Phase 4: The final phase was wrap-up and reality check: Does it look like it's complete? Are the terminology, styles, colors, and layout all consistent and easy to understand?

"In final drop, we pushed to a UI freeze so that we could do our localization and translation work," Ries says. "ProtectionPilot was released simultaneously in five languages."

This final phase and UI lock occurred three months before the software's production release. "We weren't fuzzy at this point; it was all there," Ries says.

"This stage was all about finding bugs, rather than validating the UI. We could focus on performance and defects; it became a QA process rather than 'What do you think about our product?'"

Even with the best-laid plans, not everything works out perfectly. Ries says the final drop of ProtectionPilot came two days before Christmas, "so the product didn't get a lot of testing until post-Super Bowl." ➔

UI tip #5:

Never shut the product down. Despite the structure of a four-phase development plan, McAfee wanted maximum feedback; therefore, they made sure ProtectionPilot prototypes were available to anyone, anytime.

"We kept the product running and operational the whole time," Ries says. "We demo'd it to everybody who walked by. Everybody could demo it, and everybody was encouraged to. By playing with it so much, we got used to how the product felt.

UI tip #6:

Let user demand defend against code creep. "Our team strove to understand what is necessary versus what's desirable," Ries says. "Everything you add in to the product affects documentation, support, and code.

"With every feature, you have to make a design decision. We tried to make reasonable and good assumptions about setting limits; we actively eliminated options, and then validated those options with the users.

"We ran into lots of features that are good ideas, but they were clipped until we can get enough evidence that there's enough demand. There are always next versions that we can use to include them."

> "Too many developers confuse what is possible with what is needed. You've got UI products out there with 200 different settings; as a user, you're not sure what to touch. It's like sitting in the cockpit of a 747."
>
> *David Ries*
> *development manager*
> *McAfee Inc.*

UI tip #7:

More is nearly always less. Related to the problem of code or feature creep is the tendency among software developers to put every possible option into the UI—a common mistake, says Ries.

"Too many developers confuse what is possible with what is needed," he says. "You've got UI products out there with 200 different settings; as a user, you're not sure what to touch. It's like sitting in the cockpit of a 747.

"It's difficult, because some users will always request more. I think you have to decide what you want the product to do, and take a stand. Among the products I've used that don't work well, most made the assumption that if they gave me more options I'd be happier.

"It's really hard to get rid of something once it's in. Yes, occasionally you're going to get the gold-plated feature—but you have to ask around, and be sure to include marketing and sales when you're making these decisions."

Part of the reason for jam-packed UIs nowadays, says Ted Olson, mile7's creative director, is the newfound freedom developers find in Web-based apps.

"People are coming out of that MSDN world, and what happens in the transition to the browser world is that pretty much anything goes, as long as you're staying in two dimensions. Given that freedom of the Web and the ability to throw out the yoke of the MSDN guidelines, there's a tendency to overdo—to overpopulate the page.

"With some clients—not McAfee—we'll go in to design a set of icons, and they'll want an icon for every single thing. We try to reduce the set of nouns to the lowest common denominator.

"We try to build very limited libraries and deploy them with care and augment them with simple descriptive text, so that the icon becomes a way-finding point and the text becomes much more descriptive—for example, title text or text on the page.

"Even though we look at industry standards, we can still make them your icons. McAfee, for example, had a color palette, and we worked hard to play to that. We based their icons on a blue cast, so that we had a nice contrast to their corporate red."

Good UI design isn't just about icons, of course; it's about the entire "canvas," the screenful of information you present every time your software starts up. In many cases, it's more like an assault than an assist.

"We try to come back and think about ways to segment this vast array of information, to see if there's a way to put a Next button in there," Olson says. "For example, wizards are a good way to chunk or simplify—though they're not even really wizards anymore, because they're not so formalized and mechanical.

"If there's a process, the UI needs to break it into steps. Say you have to choose a rate plan for your cell phone: You have to tell me about your habits and your usage; what can I do along the way to help you, and to also build my brand, but not in a shameless way? What can I do to help ensure that you do not fail?

"A successful user interface is all about speed, accuracy, and customer satisfaction. What's hard to measure is what's pleasant about an experience. We can measure speed and accuracy quantitatively, using short test scenarios that are observed across the desk or in usability labs.

"Satisfaction is harder. You can ask those questions with multiple or graded choices, but for me it's all about listening—listening really carefully for the hints, and listening in the hallways as things are being designed."

UI tip #8:

Use your UI to give the user a sense of context. A common failure of most software UIs is that the user is unable to derive any sense of context, Olson says.

"They don't know where they are in the product or the process. You'll find this in software installation scripts as well as day-to-day use.

"What was great about ProtectionPilot is that we were able to give them a dashboard upfront. When you open that product, your initial screen is a dashboard of network health, and from there you can decide what you want to do. There's always this 'home' screen.

"It sounds simple, but there's a lot of rigor we bring to these products. Once you determine that there's a lead screen that we all want to go back to, we try to add value to that screen.

"If you just stop at a user's cubicle and peek over the wall, what screen do you want to see?" In other words, what one screen will best convey the function and purpose—and value—of your software? A good UI ensures that you answer that question correctly.

UI tip #9:

Don't offer direction, and never assume. To keep UI design valid, it's important that users' experience is as close to reality as possible. Most customers won't have you at their side the first time they use your software, so your early tests should reflect that fact. In other words: observe, don't direct.

"The most helpful thing you can do is to not interfere with their process," Conrad says. "That's where you'll understand what their issues are.

"Hearing them struggle, watching them try different things—those are important. We will say, 'What are you thinking?' so that we can understand why something is not clear to them.

"We may give them a hint to try to guide them, rather than specific instruction. For example, let's say they're looking at a page with a bunch of information, and some of the links are hot, and their task is to drill down. They're looking around the page and can't find the right links; we might say, 'Where would you look for information related to that?'"

UI tip #10:

Resist the urge to make a quick fix. What you do with this information is key, says Conrad. There's a tendency to "fix" user confusion with a simple pointer or a quick software update, and that may be exactly the wrong answer.

"Sometimes they'll be having a problem with a particular piece of the application, and rather than issue a quick fix, you need to step back and think about whether that's the right way to display the information or have the product function.

"Users will say that if you did this a different way, that would fix the problem—but you need the ability to not just listen to what they say, but to rethink how you're presenting the information.

"You can't assume anything. You can't assume that the users are going to think like you do, and you can't assume that a simple fix is the right fix.

"Part of the role we play is that we're not so close to the product. We're part of the team, but we don't bring the same agenda. A good UI engineer is very good at not bringing his or her own assumptions to the table; they can really look at things with a clean slate."

Here's another example:

"We were testing the landing page of a software product," Conrad says. "Clearly, users were having issues. They kept giving feedback on how we could rearrange the page or use different wording to make the page better.

"It would have been easy to make changes they suggested. However, the real message was that the users were looking for a level of information that just wasn't on the page.

"What they really needed on the landing page—but couldn't or didn't articulate—was a summary of top-level information. We made this change to the UI and tested it with the users. Result: problem solved!"

(Continued on page 14 ➔)

UI tip #11:

Treat UI as an ongoing program, not a one-off. At McAfee, they call it their Joint Development Program—abbreviated to "JDP" in internal usage.

McAfee budgets time and development personnel to manage the JDP and react to user feedback, and they involve their external UI design team every step of the way.

"McAfee's JDP is so crucial, because we get to test all along the way," says Olson. "It gives us the platform for iterative testing from the concept right through to the pixel-based design.

"By the time we're at the graphic design stage, we're designing task-based usage scenarios, so we're able to observe them using the actual or a mocked-up product, and we can gather all that feedback and roll that up into the design. This is not a one shot, it's a consistent program.

"They pull a JDP call together, and they get people from Helsinki, Toronto, Beaverton, all at one time. They have their high-stakes users in this process, people who live and breathe using these products.

"Yes, they're fans of McAfee, but you know these people: They're IT, and they'll drop your stuff like a hot rock if a security product fails and a virus crashes 300 computers.

> "As long as the user group selected for a usability test is a good sample of the product's user base, the tests will provide valuable data,"
>
> *Pam Conrad*
> *project manager*
> *mile7*

UI tip #12:

Get a manageable but representative group. What's the right number of users to include in your equivalent of McAfee's JDP? "The best sample is going to be larger, but it has to be manageable," Olson says.

"Someone has to set up those calls, keep the JDP people happy, and walk around with the clipboard. "I think four to five at any one time is about all you can reasonably handle and still get valuable input.

"You need people who are willing to be vocal, and you need a variety across the spectrum of companies you're trying to hit; you want a mix of experts of newbies."

But if you're looking for the widest possible input, why not go to many more users?

"Well, first of all there's the matter of expense," says Conrad. "As long as you're careful about whom you're getting and you really know your audience profile, I think you can get a really good representative sampling with 12 to 20.

"As long as the user group selected for a usability test is a good sample of the product's user base, the tests will provide valuable data."

"You're testing user performance. Typically, there isn't that much variation in user performance. So, working with 12 to 20 users will provide the necessary UI feedback.

"This is different from market research, which is typically opinion-focused; that requires polling much larger numbers of people because there can be such a wide range of opinions."

UI tip #13:

Choose active, willing, and unbiased users. For the ProtectionPilot development program, McAfee went to a dozen users. "We involved them very early, before we started writing code, and we kept them actively involved," Ries says.

In addition to the time spent playing with the prototypes, users were asked to take part in an hour-long phone call once a week over a 12 to 14-week cycle.

"We worked with our sales organization to create a profile of the kind of company we were looking for. We wanted people who were actively managing antivirus, but not using our existing software; we didn't want to bias the results with those who were using our enterprise tool."

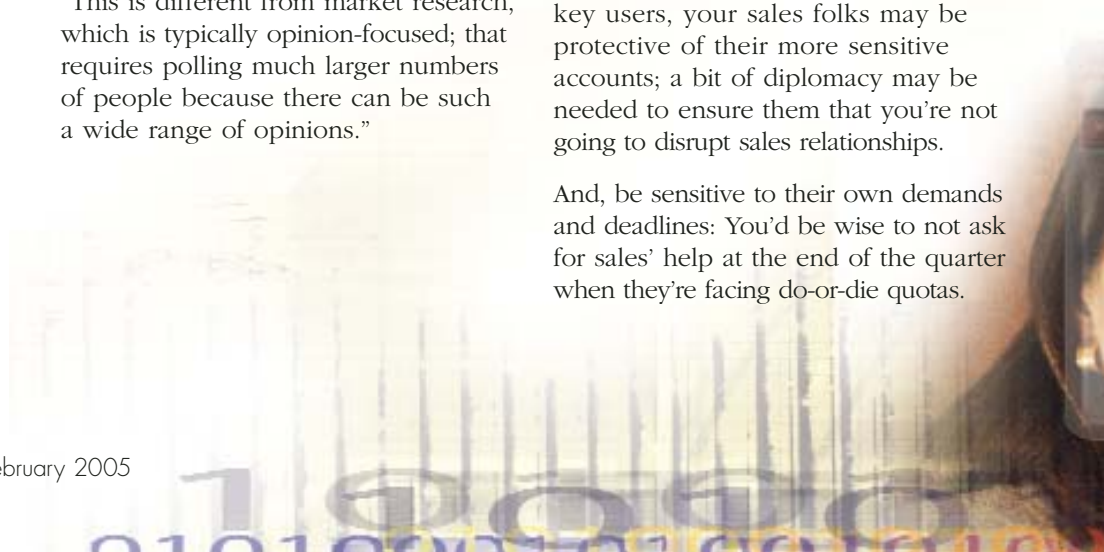
And, practicality demands that you try to find some users who are nearby. "We also wanted a couple who were local, so we could easily observe them using the tool," Ries says.

Still, McAfee's fan base is strong enough that geographic proximity isn't always required.

"We have customers who'll pay their way to get here for a local meeting," says Wyman. "We'll put them up once they get here, but it's their travel dime. They'll come because it's important to them."

One note of warning: Although your marketing department will likely cooperate with your efforts to identify key users, your sales folks may be protective of their more sensitive accounts; a bit of diplomacy may be needed to ensure them that you're not going to disrupt sales relationships.

And, be sensitive to their own demands and deadlines: You'd be wise to not ask for sales' help at the end of the quarter when they're facing do-or-die quotas.



UI tip #14:

Reward your early adopters by acknowledging their role. If you choose the right users for your UI testing, it isn't likely you'll have to bribe them or shower them with gifts; they're there because they want a say in the future of your software.

"The most important thing was that we treated them as part of the development team," Ries says. "What they got was the ability to work directly with the designers and to influence what they got in the final product. They wanted to control the end result."

McAfee's test users got a free copy of the software, but most were already entitled to it anyway as part of their existing licenses. In addition, the JDP participants did get a plaque at the end of the development period—but it was almost an afterthought, Ries says.

Here's an interesting measure of the success of McAfee's program: At the end of the ProtectionPilot program, all users were asked if they'd be willing to participate in the next JDP; they unanimously answered yes.

UI tip #15:

Get your developers (and yourself) into the right mindset. Good UI design starts with a mindset adjustment, Ries says, and your entire development team has to be on board. "The trick is to really identify with the user," says Ries.

A common failure among ISVs is believing that because a UI is intuitive to your programmers it is therefore intuitive to your users.

"For example, the word 'abort' means something to developers, but it doesn't fly with users," Ries says. "If you ask the programmer if something makes sense in his product, he will say that of course it does."

But, that over-reliance on programmers causes a lot of problems, even among software companies that think they're doing the right thing. As Ries says, nobody intentionally builds software that's hard to use.

"You have to break the typical mindset of features and technology," he says. "If you take those things out of your vocabulary, and focus on the user's ability to complete tasks, it makes it a little easier."

This means you'll have to watch your own team as well as your users. "We looking at the people who are interacting with the customers," Wyman says. "Rarely, we'll have to move a developer out of the way, but we do monitor that."

UI tip #16:

Do-it-yourself design is nearly always a bad idea. For UI design issues, McAfee sought outside expertise. "We worked with design consultants who really are the pros as to colors, interface—how you present things in a coherent way," Ries says.

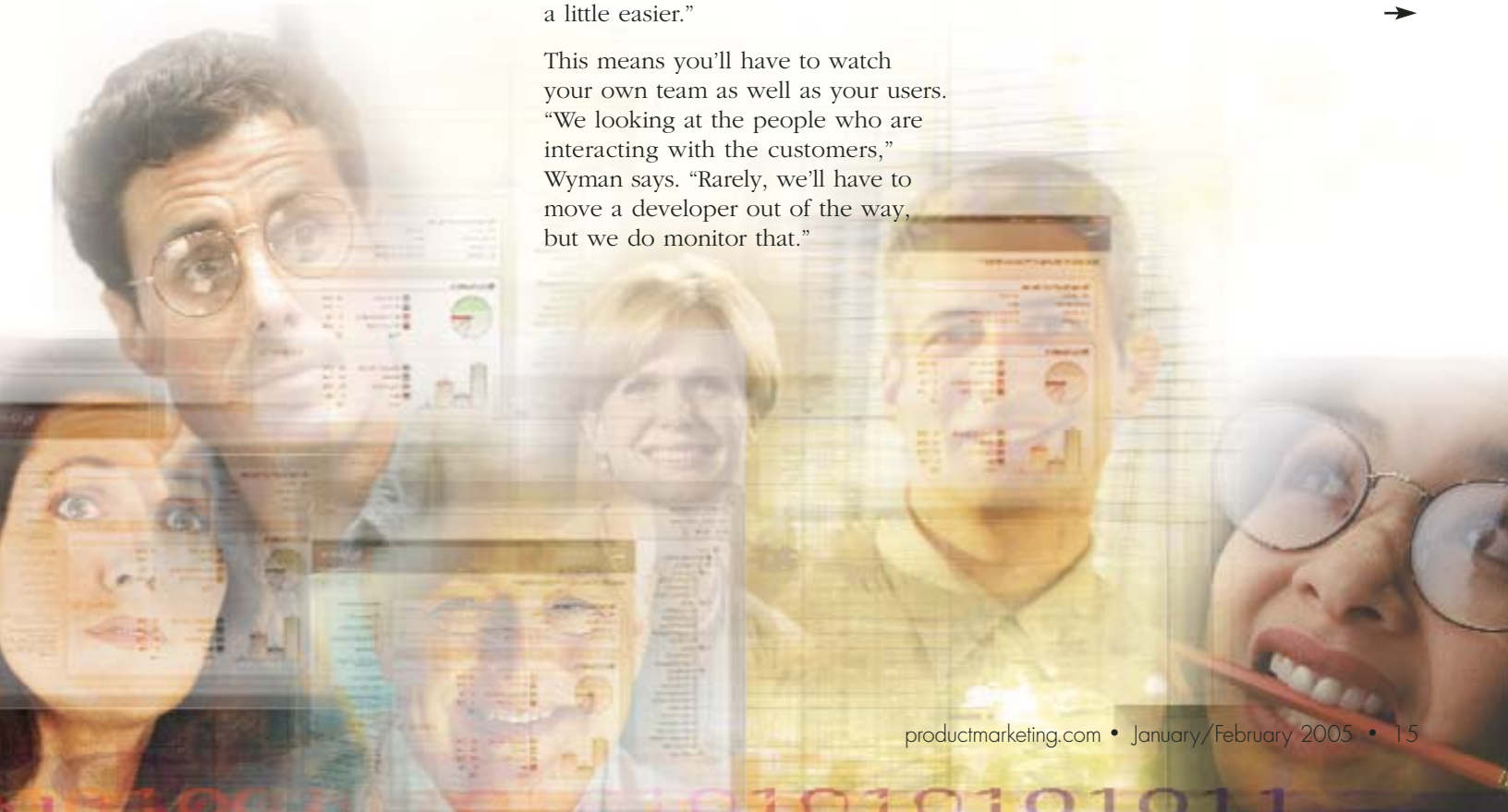
"They didn't just implement the concept; they showed us a lot of ways we could improve it." It's a big—and common—mistake to confuse programmers with artists, Ries says.

"Your programmers can maybe draw bitmaps, but that doesn't mean they're very good," he says. There's a huge difference between a professional graphic artist and a programmer who happens to know how to draw.

"A professional really does make a huge difference in the quality of a finished product. It's like the difference between a BMW and a Hyundai: Fit and finish count, and it affects sales as well as usability.

"A product that is comfortable and enjoyable to use gets used more. If you want to stand out, use pros. It will cost you more to not use them."

→



UI tip #17:

Make the designers a part of the development team. "McAfee is such a good example of what's good about the process," Olson says. "If there's one overriding theme, it's the relationship of the design firm not only to the client but also to their customers.

"We've helped them to understand the iterative design process; it's not about prima donna designers going away and designing in a black box and coming back with gorgeous designs.

"McAfee brings their very steep, deep, amazing industry knowledge and passion—they're just so into it that you can just feel the hum of energy when you walk in there.

"We bring a fresh set of objective eyes. We come in and ask why; we're not just sitting there taking direction. In a sense, we help craft the product. I don't want to take too much credit there, but there is a balance of experience that plays well.

"We become an extension of their team. We have access, up and down the team, from managers to the programmers who are tapping out the HTML for the skins.

"We don't have to fool around with all the barriers that are usually associated with a big company like McAfee. On our way in to see David, we're solving problems in the hallways.

"We started with them doing small things—just doing icons. We've now done 17 or 18 projects with them, and over time we've developed trust. Trust happens over time, and over delivery."

UI tip #18:

Don't let technology overpower usability. "A common mistake in the PC world is allowing technology to take the lead," says Conrad, "forgetting that the person still needs to be able to use it.

"If it's not easy to navigate, not easy to understand, there's going to be a disconnect there. One of the things we see is that instructions are not clear or incomplete. Is there some line that tells you what to do? Is the UI clear enough that you know what to do on a certain page? Is everything laid out in a way that matches the way people think?

"We're often brought in when engineers have done a great job on functionality, but looking at the page and knowing what to do is a whole different ballgame. We take that page and translate it into something that's easy for the user to understand. Ideally, it is a very iterative process—all the way from the front end and conceptual design, defining what the product is and what it's supposed to do.

"If there's ever a clash, it will most likely come from engineering—they don't always see the value of the UI. They see things in terms of functionality. As far as they're concerned, 'Look, the users can do all this stuff, because we've built the functionality into the software.'

"They know the product so well, they don't understand how others would have problems. We try to win them over by showing respect for what they've done, and when we put our designs up, we highlight their good work."

UI tip #19:

Bring the engineers into the UI calls. "The other part of this is that if you have a JDP program like McAfee's, get the engineers involved in those calls," Conrad says.

"You want to get the users excited about the functionality, while at the same time letting the engineers understand where the users have problems. Sitting in on usability calls is one of the best educations they can get. Lots of time things will come out in those calls that no one has ever questioned—users just get stuck.

UI tip #20:

Find an internal UI champion. "McAfee has an internal champion in David Levine," says Conrad. "He has this amazing ability to sit in meetings and interpret what engineers are saying, what users are saying, and what the UI people are saying.

"We do a certain amount of that, but he's internal, and he's there everyday. Having a person like that—someone who understands UI but can also speak the engineers' language—he's really been key in making this work. Incredibly articulate guy.

"To hold fees down, bring the pros in early, get some good advice about what this thing is going to look like, and find a way to phase your designers into the project."

Ted Olson, creative director mile7

UI tip #21:

Contain costs by including the designers early. Expert assistance on UI design will typically cost from $10,000 to $50,000 for a single software product, depending on the depth of usability and testing you want.

But the surest way to drive up the cost, Olson says, is to wait to ask for help at the last minute. "A lot of the time we get called in to do triage," he says, "which wasn't the case with McAfee. There we got called in early, and that's why it worked so well.

"To hold fees down, bring the pros in early, get some good advice about what this thing is going to look like, and find a way to phase your designers into the project.

"You don't want us on the floor all the time; you want us where we can do the most good. If your development team has already taken it way down the line, well, there's a natural reluctance on the part of the software engineer to concede on points.

If you've ever been around a software development shop, you can understand this: When they're facing mountains of work, and someone comes along and says, 'You know, I really wish this was blue,' you feel like murdering the guy."

UI tip #22:

For the best UI examples, look to consumer software. "Consumer products tend to have better UIs, in part because it's a lower common denominator," Conrad says. "You're forced to, because of the broad audience."

However, good UI design isn't just about dumbing things down, says Conrad. "Because products are task-focused, the UI should be driven by the needs of the specific user audience.

"Good UI design is all about understanding users and the tasks they need to accomplish. Given this, a software UI that might be effective for one user base might not be effective if applied to a product targeted at a different user base."

OK, so where should you look for good examples? In addition to McAfee's products, Conrad likes Adobe: "With Adobe's Creative Suite, moving between products feels seamless. Other designers I've talked to concur: Adobe understands the needs of graphic designers."

UI tip #23:

How to tell if your UI might need work. If you think your software's UI design might need help, there are a few simple things you can do, says Conrad:

Check your tech support call logs. Are there common themes and complaints? In general, are you getting too many calls? Talk with your users. Talk with prospects. What do they say? Can they accomplish the tasks you put to them? Consider usability testing—the real stuff, with outside experts in a controlled and objective environment. True, this last option isn't as cheap as the first two, but if your experience is anything like McAfee's, it could be an investment with fast and significant payback.

pm.c

A veteran of three startups, SoftwareCEO Inc. founder Bruce Hadley spent 20 years in software marketing, sales, and operations. One of those companies went public, and the other two were sold to much larger software firms. Before founding SoftwareCEO, an online resource for software executives and entrepreneurs, he was editor of one of the industry's leading news and research publications.

---

softwareCEO.com℠
PAGE ONE FOR SOFTWARE EXECUTIVES