

## Training Day

2/14/2005

By Stephen Swoyer

Used to be, training was something you looked forward to. It was a chance to refresh old skills, pick up new ones, or, in some cases, learn entirely new programming methods--all at the company's expense.

These days, an increasing number of programmers dread employer-mandated training. Take Theodore Woo, a Forte developer with a prominent technology services firm. "For me personally, I would say about half the training I've taken has eventually proven to be useful," says. "The other half, it was a case of somebody thinking it would be a good idea for everybody to know J2EE."

It's these cross-training classes that most bug Woo. In some cases, he says, he's frittered away a week's worth of his time in classes that--while interesting--are only tangentially related to his own technology domain. "My previous employer was a networking service provider, so someone got the bright idea that everyone should take a class on networking fundamentals," he explains. "It was an interesting idea, but it took me away from my current project, which was already behind schedule. Besides, I already knew what a router was. I already knew what a switch was. I already had a good grasp of networking fundamentals."

Unfortunately, says independent programming consultant Charles Martin, this type of situation sometimes happens. In fact, Martin reports, he just had a similar experience with a group of J2EE programmers.

In this case, training was subcontracted out several times over, such that by the time Martin arrived on the scene, he was six levels removed from the client. He came prepared to teach a class in J2EE fundamentals, he says, and was surprised when the assembled programmers reacted in a hostile fashion.

"The agreement was to teach a RAD course, using WebLogic Workshop, which inherently is coupled to BEA's WebLogic Application Server, and is based on Struts as a way to build the JSPs and page flows," he writes. "It makes the assumption that the students are new to J2EE, although not completely unsophisticated in the ways of Java programming."

As it turns out, however, most of the students were already well-versed in J2EE. In fact, Martin says, they had a very different understanding of why they were attending the class: "They were using JBuilder and had no intention of changing, [and] they couldn't use Struts because they had their own page flow framework, and they were using iPlanet, but were planning on moving to WebLogic because someone over their heads had made that decision for them."

After a couple days of teaching, says Martin, the hostility factor was high. So--prompted by an e-mail nastygram--he staged an intervention of sorts, explaining to the assembled coders what he had been contracted to teach, and asking them in turn what they wanted to learn. This led to a breakthrough, says Martin: The programmers mostly had questions about administration on WebLogic e.g., how they could move their iAS applications to WebLogic, how they could tune their applications on WebLogic, and he was happy to supply them with answers.

In most cases, Martin says, it wouldn't have come to this. "Unlike my usual practice, I had someone else who introduced me and went through a lot of preliminary stuff that normally would be under my control; as a result I was shy about pressing the students as much as I could have," he says, adding: "You normally try to set their expectations by saying what is and isn't in scope at that time, but if the mismatch is large enough, expectation setting doesn't help."

Priya Patel, director of marketing with Exoftware, an agile programming consultancy based in the EU, says his organization typically does several things to ensure that the content of a course comports with the expectations of students. Common practices include a "hopes and fears" exercise in which students are asked to write down their expectations with respect to the course; daily feedback, in which students are asked to comment about what is working and what isn't; interaction, in which instructors encourage students to voice their concerns with the course; and final feedback.

"We spend a lot of time talking to customers who hire us and even the folks who will be taking the course, before the course even starts, to ensure we understand their specific issues and goals," he concludes. "This is done gratis because it is an essential to getting the course right on the mark for people, plus it ensures our customers come back to us again and again."

Even so, Patel acknowledges, problems sometimes do occur, especially when not all programmers have the same skill levels.

"Of course there are always those who are more advanced or less than others in a course. The best way to deal with that is to ensure each course has some informal or downtime where people who want to get a bit more in-depth or discuss issues outside of the class scope have the opportunity to do so," he says. "We do that through breaks, lunches and [even] drinks with the class. This is often when the best material gets taught."

The key, Martin agrees, is to foster a high degree of interactivity between teacher and students. In his most recent training experience, Martin confirms, he was at a disadvantage because he wasn't involved in the training preliminaries. As a result, the assembled students didn't feel as free to voice their concerns to him. "If you have particular expectations, and it doesn't sound like it fits with the class, bring it up really early, and bring it up with the instructor," he urges.

Stephen Swoyer is a technology writer based in Athens, Ga.