

Do You Learn Just in Time or Just in Case?

Warren Harrison

The need for continuing education and training is well accepted in virtually every occupation. Today, things change far too quickly for the knowledge in any profession to remain static. New tools, technologies, regulations, and laws continually appear and evolve. Nowhere is this more evident than within the software development industry. New languages, methodologies, and computing platforms continually appear that weren't even dreamt about when many developers received their undergraduate degrees.



Most software professionals are interested in learning new things, and companies often use a promise of support for continuing education through tuition reimbursements as a hiring incentive. Of course, employers fully recognize that after six months of working 60-

hour weeks, few employees will want to take advantage of the opportunity, since it usually means spending another eight to ten hours away from their family each week.

Despite what many employers say publicly, they actually have little incentive to encourage employees to take advanced degrees or otherwise get involved in continuing education. If a company wants employees to learn a particular review technique or programming language, it will make sure they learn exactly what they need, exactly when they need it. This is called "just in time" training. However, far fewer companies are interested in "just in case" training.

Other professions and continuing education

Some professions have decided that having their members maintain currency is in society's best interest. So, they compel currency through continuing-education requirements. For example, the American Institute for Certified Public Accounting requires its members to obtain 120 hours of training or continuing education every three years. And in Oregon, my home state,

- Chiropractors must complete 20 hours of continuing education every year.
- Appraisers must receive 14 hours of training every year.
- Real estate agents must complete an average of 15 hours of continuing education every year.
- Private investigators must average 16 hours of continuing education annually.
- Police officers must average 28 hours of training every year.

In every one of these cases, practitioners must be licensed or certified by a government or quasi-governmental board to do business, and mechanisms are in place to certify courses and track individuals' training. Because their employees must have this licensing or certification to perform their jobs, it's clearly in the employer's best interest to support these continuing-education requirements not only by providing tuition reimbursements but also by making time available to take the courses.

Certified software developers and continuing education

The IEEE Computer Society has organized

DEPARTMENT EDITORS

Bookshelf: Warren Keuffel,
wkeuffel@computer.org

Design: Martin Fowler,
fowler@acm.org

Loyal Opposition: Robert Glass,
rglass@indiana.edu

Open Source Software: Christof Ebert,
christof.ebert@alcatel.com

Quality Time: Nancy Eickelmann,
nancy.eickelmann@motorola.com,
and Jane Hayes, hayes@cs.uky.edu

Requirements: Suzanne Robertson,
suzanne@systemsguild.com

Tools of the Trade: Diomidis Spinellis,
dds@aueb.gr

STAFF EDITORS

Senior Lead Editor
Dale C. Strok
dstrok@computer.org

Group Managing Editor
Crystal Shif

Senior Editors

Shani Murray and Dennis Taylor

Staff Editor Editorial Assistant
Rita Scanlan **Brooke Miner**

Magazine Assistant
Hilda Hosillos, software@computer.org

Art Director
Toni Van Buskirk

Cover Illustration Technical Illustrator
Dirk Hagner **Alex Torres**

Production Editor Production Artist
Monette Velasco **Carmen Flores-Garvey**

Executive Director
David Hennage

Publisher
Angela Burgess
aburgess@computer.org

Assistant Publisher
Dick Price

Membership/Circulation Marketing Manager
Georgann Carter

Business Development Manager
Sandra Brown

Senior Production Coordinator
Marian Anderson

CONTRIBUTING EDITORS

Robert Glass, Molly Mraz, Keri Schreiner

Editorial: All submissions are subject to editing for clarity, style, and space. Unless otherwise stated, bylined articles and departments, as well as product and service descriptions, reflect the author's or firm's opinion. Inclusion in *IEEE Software* does not necessarily constitute endorsement by the IEEE or the IEEE Computer Society.

To Submit: Access the IEEE Computer Society's Web-based system, Manuscript Central, at <http://cs-ieee.manuscriptcentral.com/index.html>. Be sure to select the right manuscript type when submitting. Articles must be original and not exceed 5,400 words including figures and tables, which count for 200 words each.

a professional designation called the Certified Software Development Professional (www.computer.org/certification). The CSDP program requires 30 professional-development units every three years. PDUs are drawn from several sources:

- Traditional education and training activities: 0.3 PDU per contact hour
- Publishing and presenting articles: ranging from 20 PDUs for publishing a book to 0.3 PDU per hour for preparing and presenting a conference paper
- Simply doing your job: 3 PDUs for every 1,500 hours of work

So, the CSDP requirement translates into about 20 hours of continuing education each year.

Nevertheless, the concept of licensing or certification among software developers hasn't caught on. For the vast majority of the world, software development is still basically an unregulated profession.

How do we stack up in practice?

These 20 hours work out to about 23 minutes a week, or 100 minutes a month. However, even with this modest amount of time, most software employers seem to have difficulty sustaining this level of professional development for their employees.

I recently came across an online poll that asked programmers about the annual number of training days they receive (www.programmersheaven.com/c/userpoll/Poll_archive.htm?PollID=136). Here are the results:

- None: 40%
- 1–2 days: 12%
- 3–5 days: 8%
- 6–9 days: 5%
- 10–14 days: 7%
- 15–30 days: 5%
- > 30 days: 15%
- Don't know: 9%

Although it's very likely unscientific, I have no reason to believe that the poll is particularly inaccurate (the percentages

don't sum to 100% because of rounding errors). In short, it suggests that over half the programmers in the field today get two or fewer days of training annually—and most get none at all.

Computerworld ranked their 2003 list of the “top 100 places to work” by the annual number of training days per IT employee. Wal-Mart came out on top with 26 training days (impressive!), whereas the bottom eight companies provided two or fewer. Remember, these were the 100 best places to work. The annual training budget for these 100 companies ranged from US\$152 to \$6,648 per employee. I don't even want to think about what the companies that aren't considered the top places to work invest (or rather, don't invest) in training.

The results, while not particularly surprising, raise serious issues for us in the software development industry, which changes as quickly as it does. Especially in light of the continuing-education requirements of many other professions that likely have a lower level of knowledge volatility than software development, the weak showing by software developers' employers is unsettling.

How can we improve?

Two issues are at the heart of our field's lack of support for continuing education. The first is general reluctance among many software developers to invest in additional training. Software professionals need to understand that to stay competitive, lifelong learning is a fact of life. Of course, if you're reading this, I'm preaching to the choir. You've already decided to keep up with the field, and you probably put your money where your mouth is by being a member of the Computer Society.

The other issue that affects training and continuing education is companies' reluctance to invest in it. I've heard it said that as the software economy has shrunk over the years, less money is available for training and education. This brings to mind an oft-quoted observation you'll hear among advertising types: “When times are good, you can afford to advertise—and when times are bad, you can't afford not to advertise.”

Clearly we can say the same about training our workforce. As the economy worsens, training and skill enhancement become that much more important.

The peer-based in-service training model

One approach taken by many organizations whose employees must maintain continuing education is based on the model of scheduled in-service training delivered by peers. This is usually much more cost efficient and more effective than the approach software companies usually take—sending employees to a distant city to sit through a one-day tutorial held at a hotel.

For example, peer-based in-service training is a common model used by small and moderate-sized police departments that can afford neither a fully staffed training division nor the cost of sending each officer to centralized training facilities. In this model, field officers become “experts” in a particular skill—first aid, defensive tactics, emergency-vehicle operations, firearms, and so on—by attending a certified instructor’s course. The rest of the department rotates through courses taught by these experts on a monthly basis. For example, one month everyone in the department might have to take an eight-hour course on defensive tactics from the in-house expert on that skill. The next month they might take a four-hour CPR refresher course from the in-house first aid expert. These courses are used to introduce new techniques and practice previously learned ones. When not occupied teaching their courses, the experts are back out on the streets with their students.

The vast bulk of software development businesses are much like these police departments. They can’t justify the cost of a complete training department or afford to send every employee to a commercial course.

You’d start peer-based in-service training by sending 10 percent of your senior programmers for intensive study in a particular aspect of programming. When they return, each would be responsible for training their colleagues in their respective specialty. This might en-

tail having scheduled training times every month, so over a period of a year every programmer in the company will have had an opportunity to have courses in up to a dozen different topics.

This model has three advantages:

- Everyone gets training every year, unlike the more traditional approach where only a subset of employees get trained owing to budget constraints.
- Because everyone gets trained, and by the same people, the level of expertise within the organization is more consistent.
- Instructors remain available to answer questions and provide advice even after they’ve given the courses.

Of course, whether they’re being trained in-house or at a conference center in a remote city, developers undergoing training must divert some time from work in order to participate. With regular in-service training, availability is more predictable and can be worked into project budgets and time lines.

Implications

This model has significant implications. The first and most significant is that you’d have to replace courses in various technology topics with courses for trainers of those topics. Learning how to do something is much different than learning how to teach it.

You’d also need standardized course material. For example, in the case of in-service law enforcement training, instructors are given training materials and tips on how to teach the material. The last thing you want is to have your instructors spending time to develop original training material that might be inconsistent with approved training materials.

value your feedback. How much training does the average developer in your company receive on the company’s nickel (including the time necessary to participate in the training)? I’d also like to hear from those of you who already practice peer-based in-service training. Please write me at warren.harrison@computer.org. ☺

EDITOR IN CHIEF

Warren Harrison

10662 Los Vaqueros Circle
Los Alamitos, CA 90720-1314
warren.harrison@computer.org

EDITOR IN CHIEF EMERITUS:
Steve McConnell, Construx Software
stemcc@construx.com

ASSOCIATE EDITORS IN CHIEF

Education and Training: Don Bagert, Rose-Hulman Inst. of Technology; don.bagert@rose-hulman.edu
Design: Philippe Kruchten, University of British Columbia; kruchten@iee.org
Requirements: Roel Wieringa, University of Twente; roelw@cs.utwente.nl
Management: Don Reifer, Reifer Consultants Inc.; dreifer@earthlink.net
Quality: Stan Rifkin, Master Systems; sr@master-systems.com
Experience Reports: Wolfgang Strigel, QA Labs; strigel@qalabs.com

EDITORIAL BOARD

Christof Ebert, Alcatel
Nancy Eickelmann, Motorola Labs
Martin Fowler, ThoughtWorks
Jane Hayes, University of Kentucky
Warren Keuffel, independent consultant
Deependra Moitra, Infosys Technologies, India
Suzanne Robertson, Atlantic Systems Guild
Diomidis Spinellis, Athens Univ. of Economics and Business
Richard H. Thayer, Calif. State Univ. Sacramento

ADVISORY BOARD

Stephen Mellor, Mentor Graphics (chair)
Dave Aucsmith, Microsoft
Maarten Boasson, Quaerendo Invenietis
Robert Cochran, Catalyst Software
Annie Kuntzmann-Combelles, Q-Labs
David Dorenbos, Motorola Labs
Dehua Ju, ASTI Shanghai
Tomoo Matsubara, Matsubara Consulting
Dorothy McKinney, Lockheed Martin Space Systems
Bret Michael, Naval Postgraduate School
Susan Mickel, Lockheed Martin
Ann Miller, University of Missouri, Rolla
Melissa Murphy, Sandia National Laboratories
Grant Rule, Software Measurement Services
Girish Seshagiri, Advanced Information Services
Martyn Thomas, Praxis
Rob Thomsett, The Thomsett Company
Laurence Tratt, King’s College London
John Vu, The Boeing Company
Simon Wright, SymTech
Jeffrey Voas, Cigital

MAGAZINE OPERATIONS COMMITTEE

Bill Schilit (chair), Jean Bacon, Pradip Bose, Doris L. Carver, Norman Chonacky, George Cybenko, John C. Dill, Frank E. Ferrante, Robert E. Filman, Forouzan Golshani, David Alan Grier, Rajesh Gupta, Warren Harrison, James Hendler, M. Satyanarayanan

PUBLICATIONS BOARD

Michael R. Williams (chair), Michael R. Blaha, Mark Christensen, Roger U. Fujii, Sorel Reisman, John Rokne, Bill Schilit, Linda Shafer, Steven L. Tanimoto, Anand Tripathi