You Can't Train That!

Or, how do you make an expert?

Rob Foshay, Ph.D.[1]

Think of who your MVP is: the person who can provide innovative, valuable solutions to your customers (whether internal or external) – not just products. The person you call when everyone else is stumped and can't fix what's wrong. The person who can get the job done in a fraction of the time most others need.

Do you really understand how your MVP's work their miracles? Probably not. Your MVP's have *expertise,* acquired over years of experience, perhaps with the aid of a mentor. What your experts know probably isn't written down anywhere: it goes well beyond what's in the policy and procedure manuals, and well beyond what's in your training (for which your best experts probably have a casual disregard). If you (or a new recruit) asks them how they do it, the experts probably can't explain it. You can watch them work, but you can't get inside their heads. If you ask them how to teach someone else, they'll tell you the history of how they learned everything they know – but that's not what you asked. For all these reasons, in most organizations, the knowledge the most valuable players have at every level is *tacit.* It takes years to build that level of expertise, so the experts are scarce. You (and your experts) may believe that what they know can't be taught, it can only be learned over years of experience, as they did.

If you believe expertise can't be taught, in a sense you're correct. Conventional methods of training can't do it, and conventional instructional design doesn't deal well with expertise. But thanks to recent research in the learning sciences, we now have a much better understanding of the kinds of knowledge which underlie expertise, and there have been major advances in our understanding of how to teach it. The research has shown that what experts are good at is *ill-structured problem solving* (problems with many ways to get to the answer, and often with many "right" solutions as well as some clearly "wrong"ones). The knowledge they have is of three types: *how it works* (the ability to predict and explain the behavior of the system they manipulate), *decision-making rules* (also called cognitive strategies), and *context knowledge* (knowledge of the system and its environment).[2] Experts' knowledge is so fully integrated that they literally can no longer take it apart or explain it. That's why an expert can usually tell you the solution, or look at a novice's work and say, "that's not how I would do it"—but that's as far as it goes. Furthermore, experts are very efficient: they gather less information than novices do, and they act based on experience and insights about the system and the context that wouldn't even make sense to novices. So in a real sense, simply taking apart what an expert does and explaining each step to a novice won't work, either.

---

[1] The Foshay Group
www.foshay.org
rfoshay@world.oberlin.edu

[2] For more on these types of knowledge, see The Foshay Group's paper, "What is Expertise?"

Researchers have developed a family of analytical techniques, collectively called *cognitive task analysis*, for systematically "unpacking" all three types of expert knowledge. The output of the analysis is a map of the expert's knowledge. Once the map exists, then it's possible to combine a range of training strategies to teach it, in a way that novices can understand.

Teaching *how things work* and *context knowledge* can be done using specific types and combinations of explanation, example, practice and feedback. These instructional strategies can be implemented through a wide range of e-Learning and classroom teaching techniques. *Decision rules*, on the other hand, must be taught using specially constructed *case problems*, preferably with learners working in groups of 4 or less, and talking to each other in a specially structured dialog. Additional case problems are needed for "worked examples" to model the process, and for use in assessing and certifying problem-solving proficiency. It is possible to implement these specially-constructed case problems in classrooms, in webcasts with audio conferencing and/or instant messaging, in online threaded chat environments typically available in learning management systems (which don't require everyone to be online at once), and even in self-instructional online environments. After this training comes an "apprenticeship" period where new trainees return to their work environments and tackle real problems with real customers, but with careful management of problem complexity and with availability of mentoring. If needed, the training + apprenticeship cycle may repeat over a period of months until the trainees have the experience (and confidence) to deal with the most demanding problems, and they can be certified as expert.

But does this new type of training really work? Research in a wide range of industries points to major gains in efficiency, often reducing the time needed to "grow" an expert from many years to a few months (one research review concluded that 5 years of experience could be compressed to 50 hours of training[3]).

Example #1: Management training

A large organization needed to update its management training to 10,000 managers at all levels, in a compliance topic. The old course required 2 days. By applying the methods of cognitive task analysis, the course was shortened to 1 day, with equal or better effectiveness. For the first 500 trainees, training required 590 person- days instead of 1,087 with the old course, resulting in a net savings of 2.5 person-years.[4]

---

[3] Means, B. & Gott, S. (1988) Cognitive task analysis as a basis for tutor development: Articulating abstract knowledge representations. In Psotka, L.D., Massey, L.D. & Mutter, S.A., *Intelligent Tutoring Systems: Lessons Learned.* Hillsdale, NJ: Lawrence Erlbaum.
[4] Clark, R.E., and Estes, F. (1996) Cognitive Task Analysis for Training. *International Journal of Educational Researcj.* 25 (5) 403-417.

Example #2: Installing and troubleshooting complex software systems

A major software company was preparing to introduce a new software system. Field engineers, who were all certified network engineers, needed to be trained in how to install and troubleshoot the new product. A conventional training analysis revealed a daunting problem: each installation of the system used a custom configuration, and there were over 350 error messages the technicians had to handle; troubleshooting each problem depended on the exact configuration of the system, so it was both overwhelming and impossible to develop troubleshooting procedures and train the technicians in them within an acceptable time (one calculation of conventional training time was *6 months* of classroom time!)

Through cognitive task analysis, a way was found to dramatically reduce the training time required. The training solution was to:

- First, teach how a prototype example system worked, emphasizing the inputs, processes and outputs of each of the 11 modules in the system, regardless of configuration.
- Then, teach the decision rules used to configure the modules into large systems across multiple servers.
- Then, teach the few decision rules for troubleshooting the system which were "new" to the trainees.

At each stage, the field technicians started with direct instruction including explanation, example, and practice with feedback. Then they moved to case problems which required them to invent their own procedures for configuring and troubleshooting, by applying the decision rules to the specific context of the customer's system.

The bottom line: training that could be done in a few hours, rather than months. As a side benefit, new releases of the system were expected to require little or no retraining: simply a field bulletin notifying the field of what had changed about the decision rules governing the configuration or behavior of the system.