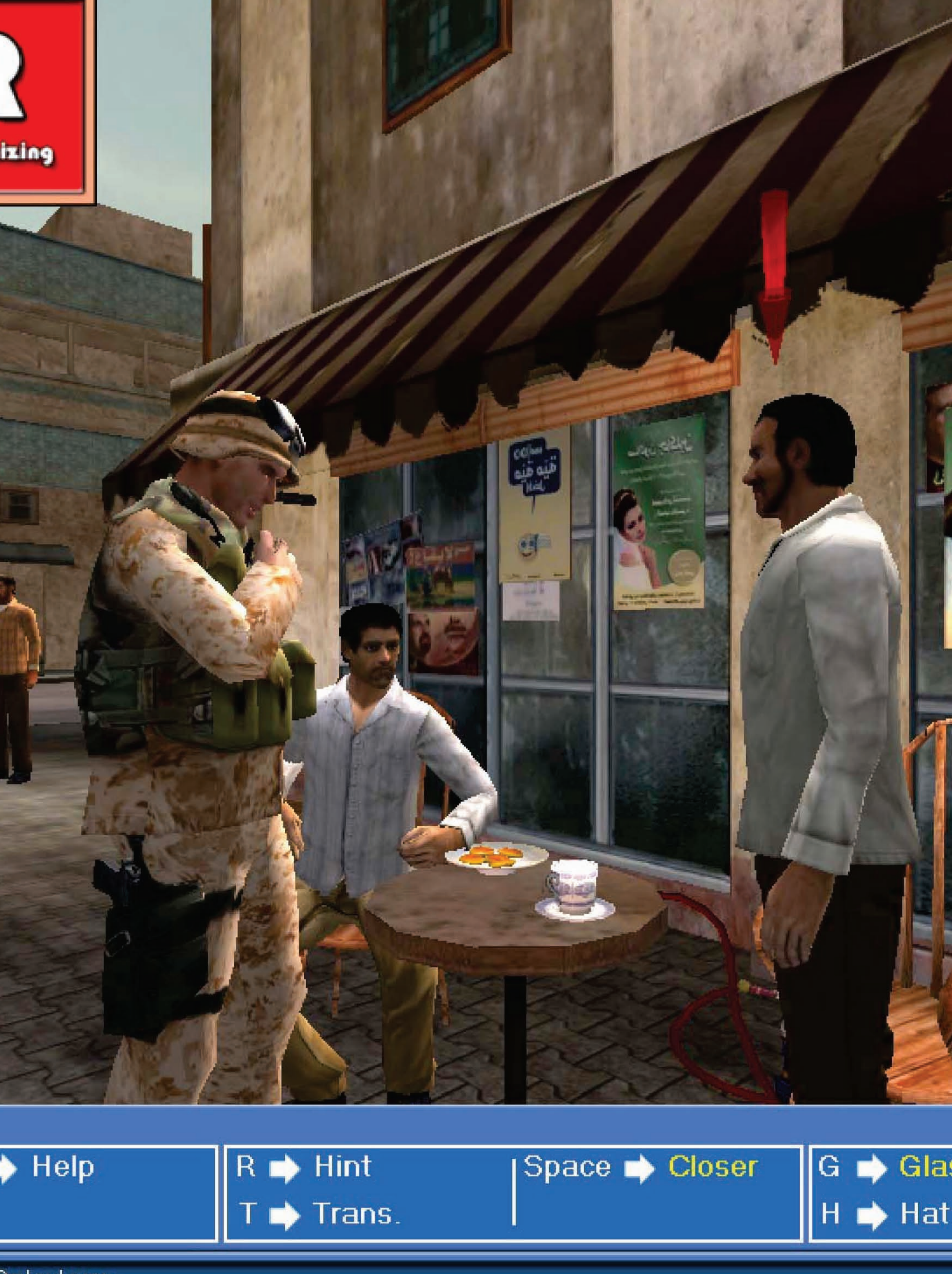
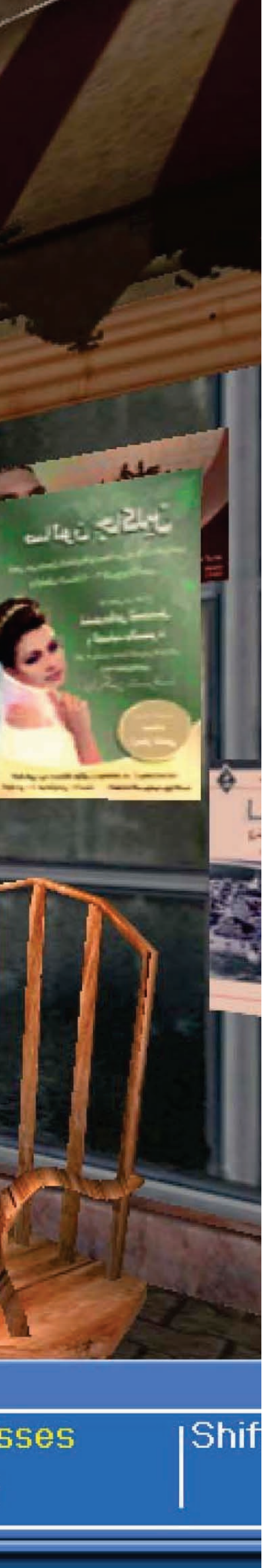

izing
Help
R Hint
T Trans.
Space Closer
G Glas
H Hat



GAMES FOR TRAINING

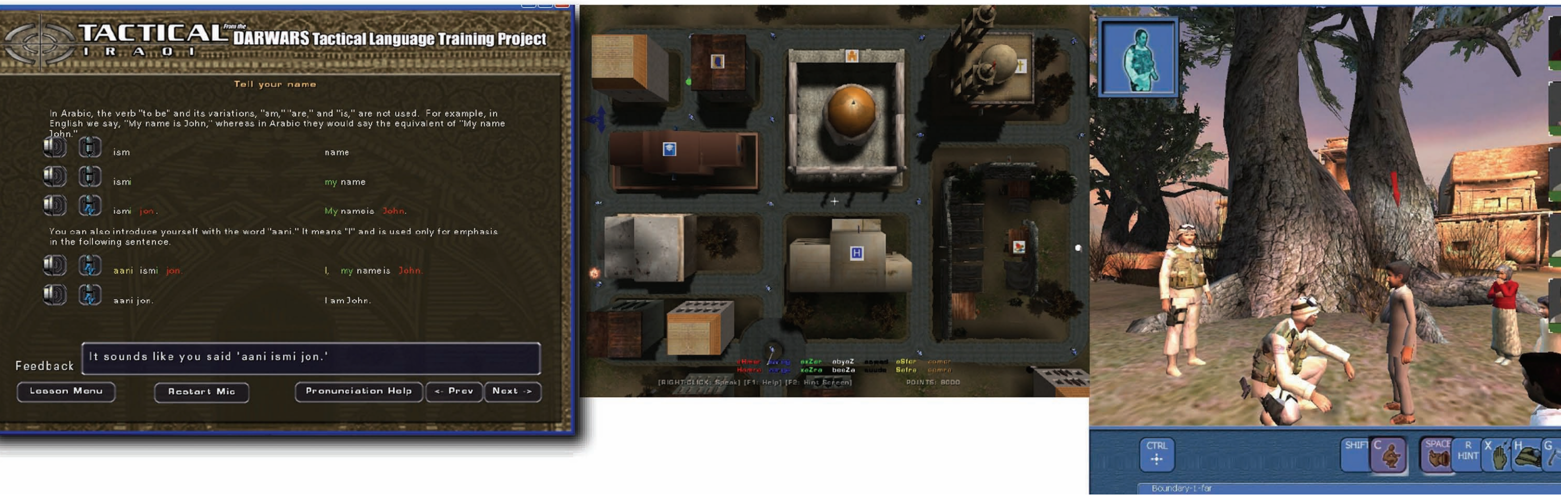
DARPA simulations teach habits of thought so soldiers respond on the first day of combat as if they had been there a week. More often than not, it's don't shoot, talk instead.

By RALPH E. CHATHAM

The notion of using computer games for military training is almost irresistible. Unfortunately, unstated, unexamined assumptions about learning and the character of and motivations behind commercial games can yield unpleasant results if the developer fails to keep the goal of proficiency gains ahead of the blind use of game technology. Here, I discuss the lessons I've learned over the past several years developing and deploying two new training systems within the Defense Advanced Research Projects Agency's DARWARS, or "DARpa's universal persistent, on-demand training WARS," program. Perhaps they will constrain the dark side of computer-based training from tainting other efforts to deliver a range of experiential learning through lightweight simulations.

The U.S. military trains millions of personnel continually as they rotate through

Salaam 'alaikum *(Peace be onto you) in DARPA's Tactical Iraqi tutor. (Tactical Language Training, LLC, Los Angeles.)*



Tell your name

In Arabic, the verb "to be" and its variations, "am," "are," and "is," are not used. For example, in English we say, "My name is John," whereas in Arabic they would say the equivalent of "My name John."

ism	name
ismi	my name
ismi jon.	My nameis John.

You can also introduce yourself with the word "aani." It means "I" and is used only for emphasis in the following sentence.

aani ismi jon.	I, my nameis John.
aani jon.	I am John.

Feedback: It sounds like you said 'aani ismi jon.'

Figure 1. Tactical Language Trainers consist of three parts: (a) Pashto mission mode; (b) a tactical Iraqi arcade game; and (c) a skill-builder screenshot.

multiple assignments and as technology and their missions change. Every individual in every unit must not only refresh what was previously learned but simultaneously absorb new lessons and techniques, many of which are exactly what U.S. soldiers and marines did not sign up to do.

The situation is not hopeless. In the 1970s and 1980s, the services, especially the Army, created what the Defense Science Board called a "revolution in training, delivering an order-of-magnitude change to unit proficiency in less than three weeks" [2]. A unit as large as a brigade, about 4,000 soldiers, deploys to one of three fixed-site combat training centers (CTCs) where they: engage in battles with a better-than-real enemy; measure the engagements objectively; and conduct no-holds-barred after-action reviews (AARs), where everyone from privates to colonels discuss what happened and what could have been done better. Effective but expensive, CTC training incurs costs for travel, logistics, instrumentation, and maintaining the better-than-real opposing force (OPFOR).

Over the past two-plus decades, this training approach has changed Army culture. Nothing today is done without AARs up, down, and across the chain of command. Lieutenants are pleased to hear and act on the advice of their sergeants and corporals. Army trainers view every training procedure as a route to an AAR; the quicker they get there, the better. Unfortunately, large units visit a CTC only once every three years, and many never do.

In 2001 and again in 2003, the Defense Science Board recommended bottling this experiential training revolution, deploying it electronically for use by more people more of the time, leading DARPA to create DARWARS [2]. Early in the program, after restricting it to PC-like hardware, the fallacious notion arose that DARWARS was about "games for training." There are cogent reasons for this almost irrepressible bias to viewing games, from massive multiplayer online ones down to first-person shooters, as instant training devices. After all, they superbly teach people to play games. Shouldn't they, therefore, be good at training people to do other things, too? The emerging answer is that this won't happen spontaneously. Mangling a metaphor, "a game unexamined yields an empty trainer."

Like other true believers, I, too, entered this realm with unstated, unexamined assumptions. Training games would, I expected, automatically be:

Cheap. To create, deploy, and maintain;
Fast. To provide instant development and delivery;
Effective. To transfer training automatically to competence in the real world;
Trainerless. So users get a disc and learn unsupervised; and
Universal. To make them accessible to anybody with a PC.

I was wrong. Unless we insist on proficiency gains, not games, these assumptions will prove false. However, the DARWARS experience suggests that, tempered with a dose of reality (and adequate funding), a middle course can be steered between game fanatics and developers of massive, expensive special-purpose training systems. We now have evidence from several DARWARS projects that lightweight training simulations, at their best, can be superbly effective. Unfortunately, at the median they are awful. There is serious danger of drawing the wrong lessons from DARWARS successes unless we document what works and why, what outside the game is needed, and whether learning transfers to the real world. So here are a few DARWARS lessons about adapting game and game-based technology to training I learned executing the DARWARS program [1].

Tactical Language and Culture Training Systems

These systems are not games, but their core technologies come from games, as well as from artificial intelligence and learning theory. The project arose when I realized that the military could never acquire a massive corps of fully qualified linguists. So I asked, instead, if we could put just a little gesture, culture, and mission-specific vocabulary into the brains behind every trigger finger and military steering wheel. Given other training demands, we couldn't expect much time with individual students. I charged the project team to define "tactical language," or what militarily valuable knowledge could we stuff into foreign-language-impaired brains (like mine), and do the stuffing in two discontinuous weeks. The resulting Tactical Iraqi Language Training System, first released in February 2005, comes on a single CD, free for government use, containing ~100 hours of training (see Figure 1). Levantine Arabic and Pashto are also available free to anyone with a .mil email address at support.tacticallanguage.com; French is next.

Like Gaul, Tactical Iraqi is divided into three parts: lessons; arcade games; and a mission/game mode. The vocabulary and culture tutor listens to students' utterances and applies speech-recognition technology to assess progress. Students may also navigate through a set of arcade games by speaking Arabic commands, including directions (such as left, right, north, south, and toward the river), along with place names, color names, and military rank. The more complicated the utterance, the more points the game awards. In listening modes, trainees respond to computer-delivered Arabic commands.

The answer from the University of Southern California Center for Advanced Research in Technology for Education (CARTE) to my "two-week" challenge was game technology. The game engine initially drove only the third part of the trainer—a mission environment where students talk their way (in the foreign language) through encounters with multiple characters and sticky situations in the performance of non-shooting military tasks. In early versions, a left-mouse click would yield the pop-up message: "You can't shoot; you must talk your way out."

Good	Bad	Ugly
User authoring drives user acceptance.	Many things, like human behavior, are difficult to author.	There is no golden disc and no "trainerless trainer" that compels trainees to use it by themselves. Humans must be available to ensure effective training.
Existing game engines permit early prototyping.	Game engines are tested only for gaming. Using them for something else means being aware of undocumented features.	The developer doesn't necessarily have access to the source code. For example, America's Army resources were later dropped to avoid copyright problems.
Games with broad user communities can provide resources and technical solutions.	Use of some resources, even the game engine itself, may be forbidden over concern for network security. DARWARS uses only detached networks, an approach that is not always available.	Government contracting and commercial game business models clash. It is difficult to find a business model to give training developers incentive to support lightweight simulations, especially if user authoring is, as it should be, required.
Older games provide good-enough simulation for training on low-end military computers.	Game reality usually comes from smoke and mirrors.	Training must still be built into the product. Developers must listen to users early and often.
Broad licenses of older games can be negotiated to enable wide distribution at minimal cost to the end user—a bargain when amortized over all users.	Finding cash up front for the license deters developers with small budgets. Moreover, a single commercial game can cost $4 million to $40 million. Good training based on good content does not come cheap, either.	Up-front money notwithstanding, licenses are often worth it, but negotiating a license is often a pain.
Games can provide a stable, tested simulation engine.	Game engines don't want to share (the computer).	Distribution by diffusion demands commercial game-testing practices, before initial delivery.
Lightweight simulations can reduce the need for human trainers.	There are no trainerless trainers yet.	Comments to the left are either good or bad news depending on which half of the glass you focus on.

Games-for-training lessons.

CARTE's decision to use an existing game engine was driven by cost and successful use of the Unreal engine in the America's Army recruiting game (www.americasarmy.com). We could get prototypes up and running quickly to learn if it were possible to teach some Arabic to everyone. It also kept the program alive by enabling us to show off intermediate products to users, funding sources, and the media.

We found that shifting among training modes took too much time, leading to games-for-training lesson one: The game wants the whole computer, leaving no resources for speech recognition or videos of talking heads (see the table here). CARTE traded some of these desirable features for rapid mode shifting. Since 2005, the Unreal game engine (www.unrealtechnology.com) has functioned as an operating system for everything.

Some game engine tools were quite useful; when students had trouble recognizing changing attitudes of virtual characters in the mission (game) environment, CARTE converted the "ammunition status bar" into a "trust meter." It also used America's Army resources to get started but later dropped them due to copyright restrictions.

Other game engine features were not useful. The programmer building arcade games found that when one's avatar reached a new point in the Pac-Man-like

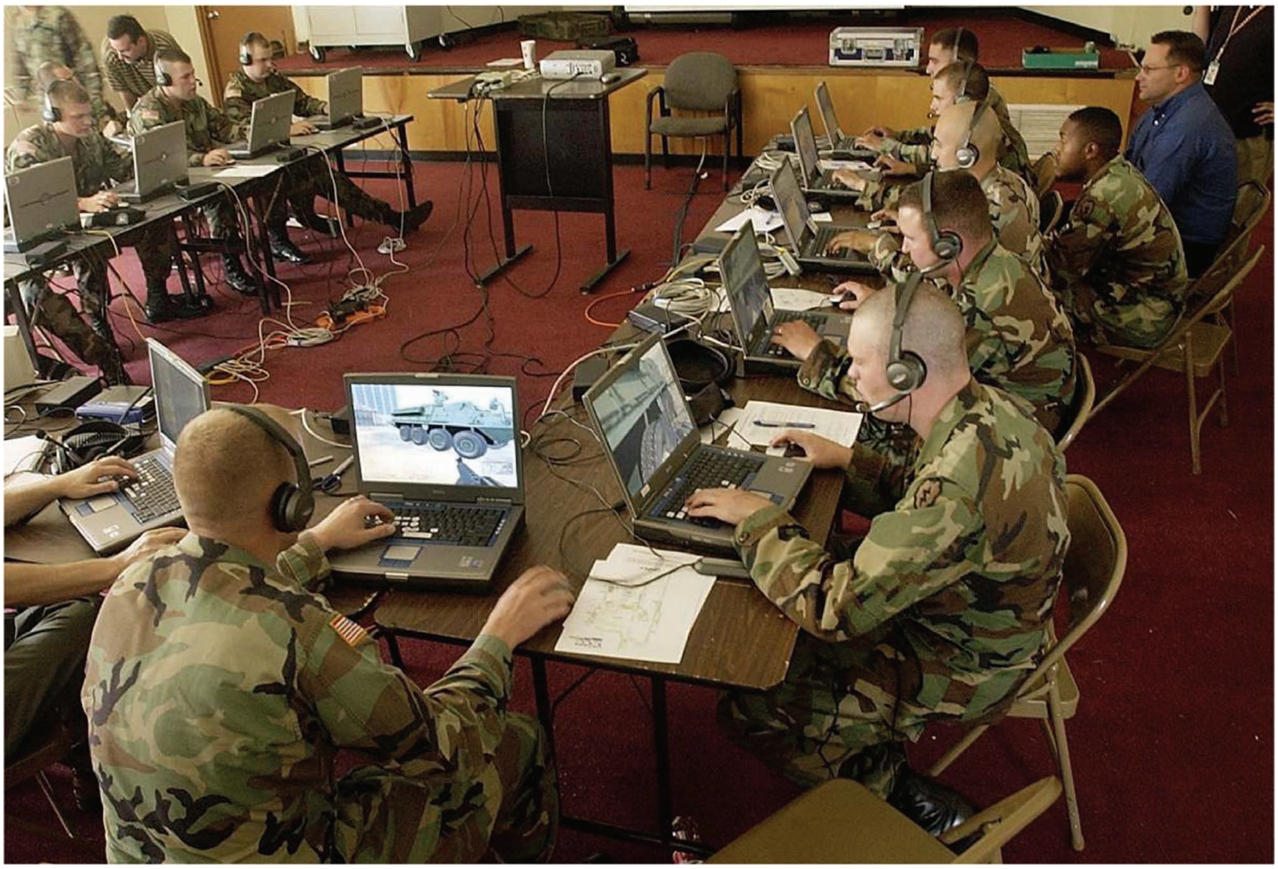


Figure 2. Soldiers at Fort Lewis, WA, training with DARWARS Ambush!, summer 2004.

maze, it assumed a random orientation rather than maintaining the direction of its last motion. The documented routine that should have solved this didn't. Weeks of poking into the dusty, disused corridors of the Unreal engine eventually led to a workaround making it possible to keep the avatar moving in the right direction. The game's broad user community helped there.

A second problem was more subtle. In late 2005, CARTE found that its new Pashto speech recognizer was not working well. Looking back at a new implementation of Tactical Iraqi—instituted when shifting from tools permitted by a research license to tools conforming to a newly purchased commercial license—it found that Arabic recognition had degraded, too. Months of searching traced this performance to an undocumented 8kHz low-pass acoustic filter. It worked well for game communications but was a disaster at speech recognition.

The lesson here is that the factors driving game-engine development may not coincide with those driving a training application. Game developers stop fixing things when the game works. This leads to dangling, undocumented features appearing when one uses the engine for something other than playing the tested game.

When a training developer moves from a low-cost research license to distribute a new training product for real, the game developer wants, justifiably, a cut of the revenue. The game engine license for the tactical language trainer alone cost more than development of a typical whole-schoolhouse training curriculum. Many military-games-for-training hopefuls balk at paying hundreds of thousands of dollars just for permission to distribute a game-based trainer. In the case of the Tactical Language Tutor family, the game license was, however, a bargain, giving the government rights to use the Unreal engine in current and future tactical language tutors, requiring (after payment of a one-time fee) only that the packaging proclaim "Epic technology inside" and that Epic receive a percentage of new development costs.

DARWARS Ambush!

Personally challenged in 2003 by the director of DARPA, I started a new program to train the voice in the back of the head of every service person in how to identify, prepare for, deal with, and recover from ambushes. I selected a development team that held training value as its highest priority. Labeling the program DARWARS Ambush!, I then set them free to build a soldier-training tool constrained only by six high-level goals:

Be guided by a respected early adopter. Col. Robert

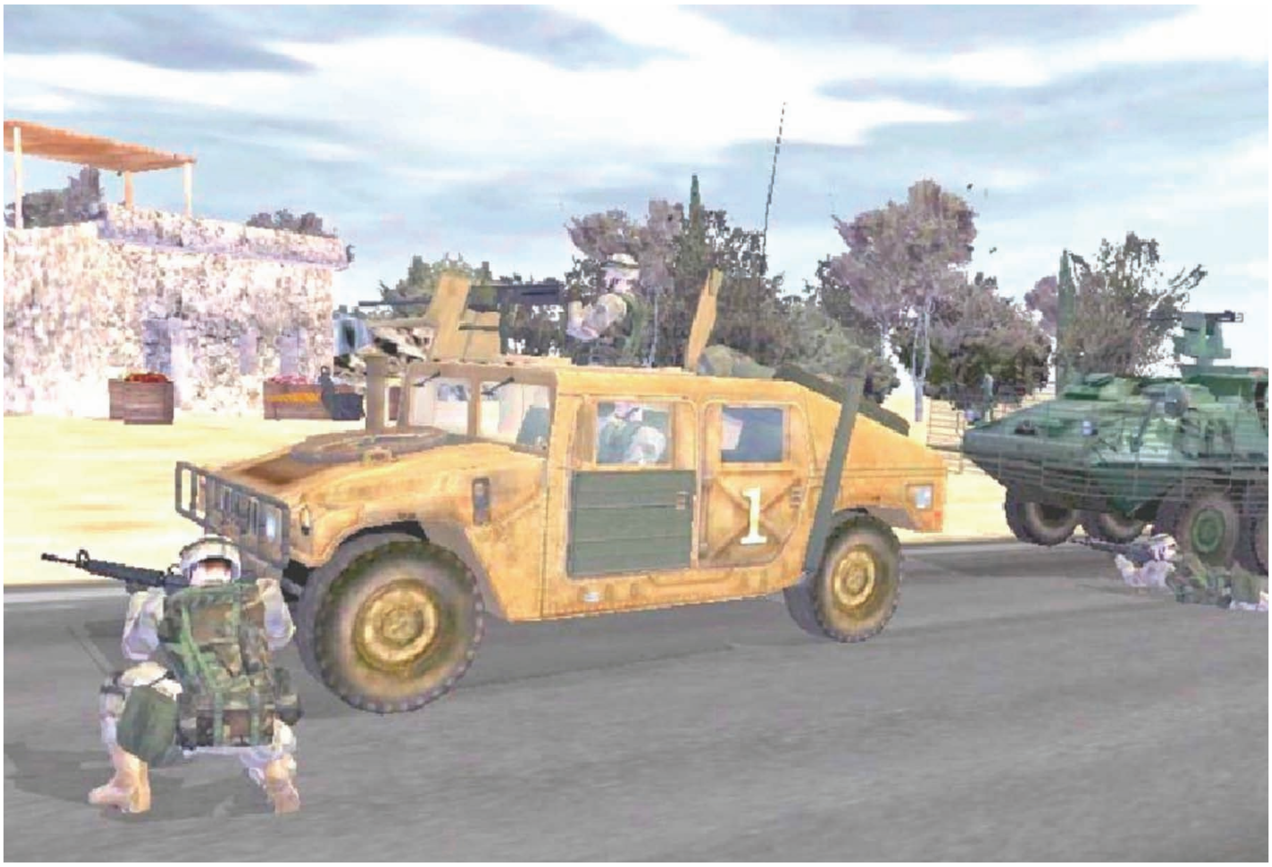


Figure 3. Screenshot from DARWARS Ambush!

Brown, commander of the 1st Brigade, 25th Infantry, enthusiastically adopted the development team and marched my contractors, including BBN Technologies, Total Immersion Software, and Jason Robar, Inc., into the mud during his CTC rotation[1];

Build for soldiers. They bear the brunt of any ambush, not the lieutenant colonels;

Provide for simple, rapid field authoring. Teach what happened yesterday and not wait months for contractor changes, with attendant costs and confusion;

Have OPFOR be the squad next door, not software, which could not adapt to a changing mission. That squad would learn as much executing an ambush as it would being ambushed; inter-squad competition might ensure that the training tools would be used continually, not played once and put on a shelf;

Use games to make training compelling, but ensure training is put inside; and

Deliver in six months.

The BBN team then exceeded these goals, creating a networked, multi-user, PC-game-based, convoy-ambush training tool (see Figures 2 and 3) that "allows soldiers and marines to experience lessons learned by others and to construct their own scenarios based upon actual experience. Trainees move about in a shared, immersive, first-person-perspective environment where they carry out mounted and dismounted operations, operate ground and air vehicles, use small arms and vehicle-mounted weapons, and communicate over multiple radio nets" [3].

This approach limited the training developer's opportunities to generate continuing revenue. The game engine, Operation Flashpoint (developed by Bohemia Interactive Studios, published by Codemasters), costs users about $10 per seat, and the six-month deployment goal tacitly assumed that the government-developed product would be free for users to copy and distribute. I had envisioned that BBN would do the rapid authoring in response to field requests. However, BBN's team took my vision a step further, embedding training to show users how to author scenarios themselves. Soldiers make all but the most complicated changes, with no cost but their own labor. Similarly, homegrown OPFOR can adapt to any new roles without developer expense.

DARPA, and others, gave BBN occasional time-and-materials contracts to add features, scenarios, or terrain. The major continuing business for the developers provides Web and phone, as well as on-site, train-

[1]Much credit for success also goes to Dan Kaufman, a program manager at DARPA, whose current goal is to develop user-authoring tools that let individual soldiers create their own DARWARS Ambush!-like mission-rehearsal tools, in a geo-specific world, in only a few days. When successful, his program, called RealWorld, will put Ambush! out of business.

THE MOST IMPORTANT LESSON OF THE PROJECT REFLECTED THE FLEXIBILITY to allow soldiers to invent and implement new ways to use Ambush! without a contractor between them and their tactics, techniques, and procedures.

the-trainer support, but this, too, doesn't represent an enticing source of continuing revenue.

I recommend against training developers selling game-based training on a per-seat basis and maintaining control over changes and charging for upgrades. Users are irked when they are unable to make even minor alterations to training scenarios. Moreover, too few users are available to hold down the per-seat costs enough to fit within unit budgets. What they can afford is their own labor.

Probably the best hope for a good product, satisfied users, and modest developer profit is what DARWARS did: provide up-front money from a central source, then pay the developer or some other entity on a time and material basis to deliver user support, set-up, and train-the-trainer services in chunks small enough to fit within a unit's own budget.

Why a Game?

The "deliver in six months" requirement demanded an existing simulation engine. We chose Operation Flashpoint as a clear winner from about 10 alternatives because it readily supports:

Outdoor terrain. Handles large-scale outdoor terrain;
Field authoring. Includes tools adaptable to local scenario creation;
Vehicles. Allows users to enter and operate vehicles;
Realism. Delivers adequate fidelity but not much more; and
Multiple players. Includes flexible multi-player capability.

Flashpoint was also a good choice for Ambush! because, at several years old, it didn't require a top-of-the-line PC with the latest video and memory cards installed. It was well-tested in similar applications by a gaming community that had created a host of simulation resources and objects available for only the cost of acknowledging their creators.

Flashpoint was also a bad choice for reasons we recognized from the start: limited support for AAR and voice communications because our training developers lacked source-code access. Also, late in the process of delivering Ambush!, a lengthy negotiation over licensing requirements erupted that took time and energy away from distribution and support activities. The problem was, in part, the result of a mismatch between the business model for making money with a commercial game, risking much money early in hopes of making it back on sales, and government acquisition practice, where much of the profit comes during a paid development period.

The licensing issues were finally resolved through an agreement linking licensing fees to actual development costs. The legal wrangling was brightened when BBN's attorneys played the game, and the technical staff got to shoot (virtually) the lawyers.

Distribution by Diffusion

DARPA sits outside normal military acquisition channels, so while our early adopter—Col. Brown's brigade and others at Fort Lewis—quickly embraced Ambush!, there was no mechanism for further distribution or even for communicating that it existed. Tactical Iraqi was in the same boat. So we engaged the press, visited units with demonstrations, and conducted word-of-mouth campaigns. Knowledge and demand for both products spread quickly, and for the next six months I received at least one email message or phone call per day asking how to get them. Mendicants received software in the next day's mail. This would not have worked had we charged the units for the government-developed tools.

Today, sites at Army and Marine bases across the U.S. and overseas have upward of 100 computers dedicated to training with Ambush! More than 20,000 soldiers, marines, and airmen trained with these tools in 2006. Had we tried to introduce them from the top, we would still be awaiting approval. Bureaucracy may be slow, but a government charge card in the hands of a major is fast.

The services have begun to pay for user support and central maintenance of the software. This is timely, since DARPA, as the attention-deficient-disorder child of defense research, as a matter of policy, abandons programs once it has taken the technical excuse away from those who say something cannot be done. Service organizations are justifiably resistant to adopting the support burden of something pushed on them from the outside. Unplanned costs must come out of an already tight budget. Distribution by diffusion can succeed only with strong user demand and demonstrable training performance.

Why Did Ambush! Succeed?

First, it met an urgent, clearly perceived need. Once users try it, they discover that training is built in. Not merely an environment in which training takes place, the disc includes: training manuals, tactically representative scenarios with descriptions of how to use them for training, and train-the-trainer information. Military billets are often "gapped,"[2] so one should not expect a first user's enthusiasm to be transferred to the next user. In late 2004, at one training facility, the new director found a stack of CDs of game-like titles all claiming to be training tools. Only the Ambush! disc told him how to train with it.

When distribution requires user acceptance, the tool must appeal to the end customer. What they adopt with their own money and effort must work well. DARWARS Ambush! gaming contractors insisted that we test intensively before release far beyond normal military training software testing, and commercial testing practice weeded out bugs that would have biased users against the initial product.

DARWARS Ambush! didn't deliver just an application. An application, alone, can lose trainees interest once they run through the initial built-in scenarios. Instead, ordinary users are able to change scenarios and create new ones to meet changing circumstances. Lessons from deployed units are incorporated into training within days. Commander-identified training needs are met through user modifications. One unit created the terrain of its home base, then used Ambush! for a disaster-relief exercise. It simulated tornado damage with a platoon of virtual tanks, knocking down trees, flattening cars, and damaging buildings. Other units used it to produce a training video of a real battle the commander used to prompt discussion of his tactical intent and explore options with his battalion.

The most important lesson of the project reflected the flexibility to allow soldiers to invent and implement new ways to use Ambush! without a contractor between them and their tactics, techniques, and procedures. Users who do these things with no delay and no cost but their own labor embrace the trainer as their own, not resenting it as imposed from above.

Finally, we listened to the users, engaging them early and often in the development cycle. We took draft systems on the road every month to discover what needed fixing. We also found that the alpha version trained units about convoy ambushes even as they showed us how to improve. The game platform allowed us to create usable training software early and thus get vital feedback quickly.

Conclusion

DARWARS Ambush! and the Tactical Language tutors both showed that a trainerless trainer doesn't exist yet but that such training tools can dramatically reduce the need for human trainers. The mere acquisition of a disc of training software seldom results in effective training. Ambush! itself reduces quickly to a free-for-all unless it is used in a setting with an instructor, training goals, and enforced AARs. Tactical Language and Culture tools, while usable by motivated individuals, deliver much more useful training in group settings. One pair of cooperating novice trainees taking turns on the same computer got more out of them than do most students with similar motivation and a computer to themselves. Human instructors also ensure that the artificialities of the simulation do not lead to negative training.[3] In fact, elsewhere in DARWARS, we have seen strong evidence of the positive transfer of skills learned in computer simulation to combat-like simulations [4].

Two game-based tools don't constitute a training revolution but do provide an existence proof of what might be done. I hope these insights lead the way to many more trainers where the gains come with the games. C

References

1. Chatham, R. *Training Superiority.* The 2005 DARPATECH Symposium (Anaheim, CA, Aug. 2005); www.darpa.mil/darpatech2005/presentations/dso/chatham.pdf.
2. Defense Science Board. *Training Superiority, Training Surprise 2001*, and *Training for Future Conflicts 2003.* Reports, Washington, D.C.; www.acq.osd.mil/dsb/reports.htm.
3. Roberts, B., Diller, D., and Schmitt, D. Factors affecting the adoption of a training game. In *Proceedings of the 2006 Interservice/Industry Training, Simulation, and Education Conference* (Orlando, FL, Dec.). National Training and Simulation Association, Arlington, VA, 2006.
4. Widerhold, B. and Widerhold, M. *Physiological Monitoring During Simulation Training and Testing, Final Report.* Virtual Reality Medical Center, contract DAAH01-03-C-R-301, July 29, 2005; www.vrphobia.com.

Ralph E. Chatham (Ralph.Chatham@darpa.mil) is co-chairman of the Defense Science Board task forces on Training Superiority and Training Surprise and Training for Future Conflicts and program manager for Training Superiority, Defense Advanced Research Projects Agency, Arlington, VA.

My thanks to Lewis Johnson of CARTE and Bruce Roberts of BBN who each created from vague ideas real training that is saving lives today.

[2]When a billet is gapped, one person leaves, often months before his/her relief arrives.

[3]When challenged with the potential for negative transfer of training at the first DARWARS Ambush! users conference in February 2006, several users countered that it would happen only if the trainers failed their students in the AAR.