Fun and Games and Software Development

Edward Grossman, Editor, *Queue*

You may recall some of the hype last year as AMD announced and then released its 64-bit processor, the AMD Opteron. What you may not remember is that one of the key groups AMD was going after with its promotional blitz was gamers. You see, at the high-end (read: high-margin) side of the PC business, power-hungry users drive the business, and more and more often those power users are gamers looking to get that millisecond advantage needed to claim bragging rights for the week.

To me this is incredible—and it illustrates the dramatic revolution that has gone on in the computer- and console-game business. Games are now a driver in the PC hardware business (CPUs, graphics cards, consoles): They're all courting gamers with more and more power. And the games themselves have kept up with the break-neck (Moore's Law) pace, keeping even the most cutting-edge hardware thoroughly taxed out of the gate (and usually obsolete a few months later).

The software has grown in complexity by multiple orders of magnitude. For the uninitiated among *ACM Queue's* readers, here's some perspective: One of the first games I ever played was *Adventure* for the Atari 2600. In that first-person-oriented game, my questing knight was literally represented on the screen as a square—a small, 4- by 4-pixel, flat, colored square. This was as much as could be mustered back in the '70s (hey, they only had 4K available for the game code). Compare that with today's nearly photorealistic characters living in 3D space, with stereophonic polyphonic digital music and sound effects, and you begin to get the picture.

For this month's special report, we've brought together industry experts to delve into how games are changing today's software development landscape. First off, regular columnist for *Game Developer Magazine* Jonathan Blow has a nice overview of where game development is currently and why it's likely harder than you think. We follow that up with Michi Henning's "Massively Multiplayer Middleware," wherein Henning walks us through his experience of being asked to implement the middleware for an online game, looking at what went well, and what didn't.

One of the most interesting trends in computer game

Computer games:

FROM "ALSO RAN" TO

"FEATURE ATTRACTION"

development is the use of multi-language approaches. Though game developers are oft thought of as coding in assembly to achieve

maximum performance, Andrew Phelps and David Parks of Rochester Institute of Technology look at why sometimes it turns out to be faster to pop from one language to another to take advantage of the strengths of each.

And, of course, any special report on game development would be incomplete if it didn't address the following two core areas: AI—as it exists in computer games, not academic research (and, *yes*, there is a difference)—is the topic of Alexander Nareyek's "AI in Computer Games." In "The Scalability Problem," Dean Macri of Intel explains why life is hell trying to write games that scale from the latest and greatest systems to your grandmother's old PIII. (It turns out there are some tricks of the trade, but they carry a price.)

We also were lucky enough to have Will Harvey, founder of There—which may end up being the closest thing yet to Neal Stephenson's *Metaverse* in his novel *Snow Crash*—sit down with open source advocate and Damage Studios cofounder Chris DiBona.

In addition to this issue's special report on Game Development, we have "Beyond Authentication," from author and Counterpane Internet Security founder Bruce Schneier, on challenges in security and authentication; "People in Our Software," an in-depth look at how human context should play a more useful role in messaging software, from John Richards and Jim Christensen of IBM Research; and a nice, opinionated rant, "The Demise of the Waterfall Model is Imminent" and Other Urban Myths," by Phillip A. Laplante and Colin J. Neill of Penn State. Enjoy! ☺

EDWARD GROSSMAN is responsible for *Queue*, so blame him if you don't like it. In earlier incarnations he was a development project manager at a still-in-business dot-com and a closet coder (his parents still don't know—"Our son Ed? Oy, he works with computers, doing something").