/dev/random
Distributed Illogic

ROBERT G. FERRELL

Robert G. Ferrell is an award winning author of humor, fantasy, and science fiction, most recently *The Tol Chronicles* (www.thetolchronicles.com).
rgferrell@gmail.com

As a teen I bought my first car (a '69 Chevy Impala Custom, in the trunk of which you could park most of today's models with room to spare) using money I'd saved from various jobs after school and on weekends. The engine was a small block 350 without any fancy electronics, emissions control devices apart from an exhaust manifold, or fuel injection mumbo-jumbo. I knew how it worked, how to do routine maintenance and simple repairs, and where everything was in the engine compartment. I could replace/gap spark plugs, change the oil, filters, and distributor cap, adjust the timing, play with the carb mixture, and so on. It was a straightforward, reliable vehicle, even if it did lack certain optional luxuries like functional motor mounts.

When I open the hood of my 2001 Trans Am (yeah, I still drive that wonderful dinosaur, when I drive at all), even after 15 years of ownership I'm frankly at a loss to understand any more than half of what I see in there. I'm lucky if I can find the oil dipstick, to be brutally honest. So much has changed in the world of automotive technology since 1974. I plug in my diagnostic computer readout thingamajig whenever I get a warning light on the dashboard, but I don't have any idea what the messages it displays are talking about most of the time. I just shrug and hit "erase all."

Technology in virtually every area has advanced significantly, of course. Take distributed computing, for example. It wasn't that long ago that the suggestion that bits and pieces of not only our data but the very software and hardware that process it would be scattered hither and yon across the Internet like propaganda leaflets dropped from a vintage South Korean Piper Cub would have been met with ridicule or at least eye rolls and head-shaking.

I figure if we're going to continue down the distributed everything road, we may as well take it to the next level and distribute the electricity that runs these machines while we're at it. So let's say I need 30 amps at 110 volts to run a server rack that's processing on the cloud. In the old system, I would simply plug into a UPS that then was fed by a circuit from the local electrical utility (in most cases). How quaint. In my sleek, modern power supply engineering paradigm, nodes all over the planet advertise the number of electrons they have sitting around unused at the moment and send them wherever they're needed on request.

At any given moment, then, you might be powering your cloud server with juice from Montevideo, Edmonton, Aberdeen, Zagreb, Taipei, Jakarta, and Perth. I'm not certain, but that could require some conversion from European or Asian volts to North American volts. I think electrons might travel on the other side of the wire in most of those countries, too, but we can work with that.

As an adjunct to the distributed computing trend, I propose we stop calling it the "Internet" and adopt "Omninet." It has a more inclusive ring, don't you think? With the term Omninet you don't need to make cumbersome distinctions like "the Internet of Things" because

Omninet pretty much covers all that ground by default. It also has a sort of Orwellian *the government is watching you* feel to it that should prove popular amongst certain elements of today's (justifiably) paranoid society.

Maybe it is also time we consider taking our distributed physical architecture to its logical extreme. Why stop at the board level when you can drill on down to individual components? We can assemble the necessary circuits on the fly from a database of hundreds of thousands or even millions of resistors, capacitors, diodes, integrated circuits, and so on available worldwide using the new generation of just-in-time hardware compilers I recently made up. That way the only piece of processing hardware you actually need on premises is the compiler itself. Everything else can be recruited from the Omninet in real time. Maybe we could even figure out a way to assemble the compiler on the fly. Closed loops are so entertaining.

Having a ten thousand-mile-long electrical bus might seem a little ponderous, but think of the local thermal advantages, not to mention the savings from not needing to buy equipment that becomes obsolete before you can get it installed and configured. Heck, I see significant advantages even for home users, especially gamers. One of the reasons I finally gave up on PCs and went to console gaming exclusively some years ago is that I got tired of having to upgrade my video card for every new astronomical-polygon count game release. Sixty bucks for the game and another three hundred for the hardware to run it puts a real dentaroo in the ol' household budget, know what I mean? I have a cabinet drawer that could supply the nucleus of a decent graphics card museum.

If the game itself could actually specify its minimum hardware requirements for running and recruit the necessary components from the Omninet, that would be just super. Sure, a few itty-bitty latency issues might crop up at first, but I'm certain they'll be overcome. After all, we can stream 4K cat videos to corners of the planet where indoor plumbing is considered a novelty. All a gamer would need is a monitor, an Omninet connection, and whatever interface was necessary to make the distributed components work together.

Of course, some might argue, why bother with on-premises hardware at all? Just use a sort of distributed Steam-like system where each player forks a new instantiation of both the game and platform. Simplifies multiplayer quite a bit. We might take clipping to new heights, as well: instead of merely hiding the parts of a rendered scene not currently visible to the player, don't even *write the code* until the predictive engine determines it will be required soon. Who needs human programmers these days, anyway? Am I right? Step away from the pitchfork, meatbag.

If I've said all this before, I'm not surprised. Three days is about the maximum I can go without repeating myself at dinner table conversation with my wife; I've been writing this column now for over ten years. You're lucky I don't just select random paragraphs from previous columns and string them together. I called it "/dev /random" for a *reason*, you know.

Um, actually, that's not a bad idea. Reusable code is all the rage, after all. Maybe I can put every paragraph I've ever written into a database and let you mash them up yourselves: sort of a literary mix tape. This sort of betrayal only hurts if you let it.

Register Now!

OSDI '16

12th USENIX Symposium on Operating Systems Design and Implementation

Sponsored by USENIX in cooperation with ACM SIGOPS

November 2–4, 2016 • Savannah, GA

Join us in Savannah, GA, November 2–4, 2016, for the **12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16)**. The Symposium brings together professionals from academic and industrial backgrounds in what has become a premier forum for discussing the design, implementation, and implications of systems software.

Co-located with OSDI '16 on Tuesday, November 1:

• **INFLOW '16:** 4th Workshop on Interactions of NVM/Flash with Operating Systems and Workloads

The full program and registration are now available.
Register by Friday, October 7, and save!

www.usenix.org/osdi16

usenix
THE ADVANCED COMPUTING SYSTEMS ASSOCIATION