A Brief History of Warehouse-Scale Computing

Reflections Upon Receiving the 2020 Eckert-Mauchly Award

Luiz André Barroso, *Google, Mountain View, CA, 94043, USA*

Receiving the 2020 ACM-IEEE Eckert-Mauchly Award this past June was among the most rewarding experiences of my career. I am grateful to *IEEE Micro* for giving me the opportunity to share here the story behind the work that led to this award, a short version of my professional journey so far, as well as a few things I learned along the way.*

THE PRACTICE OF COMPUTER SCIENCE

For many of us our earliest models of professionalism come from observing our parents' approach to their work. That was the case for me observing my father, a surgeon working in public hospitals in Rio de Janeiro. Throughout his career, he was continually investigating new treatments, collecting case studies, participating and publishing in medical conferences, despite never having held an academic or research position. He was dedicated to the practice of medicine but always made time to help advance knowledge in his area of expertise.

Without really being aware of it, I ended up following my father's path and became a practitioner myself. As a practitioner, my list of peer-reviewed publications is notably shorter than most of the previous winners of this award, but every time I had something valuable to share with the academic community, I felt welcomed by our top research conferences, and those articles tended to be well received. Practitioners like myself tend to publish papers in the past tense, reporting on ideas that have been implemented and launched as products. Practitioners can contribute to our community by looking back and showing us how those ideas played out (or not) in practical applications. Commercial success or the lack thereof can be an objective judge of the merits of research ideas; even if cruelly so at times. In giving me this award, the IEEE Computer Society and ACM are highlighting the role of practitioners in our field.

Now, as this award is about the practice of warehouse-scale computing, I should get to that with no further delay.

A BRIEF HISTORY OF WAREHOUSE-SCALE COMPUTING

If it is indeed true that "great poets imitate and improve,"[1] poetry and computing may have something in common after all. Warehouse-scale computers (the name we eventually gave to the computers we began to design at Google in the early 2000s) are the technical descendents of numerous distributed computing systems that aimed to make multiple independent computers behave as a single unit. That family begins with VAXclusters[2] in the 1980s, a networked collection of VAX computers with a distributed lock manager that attempted to present itself as a single system to the user. In the 1990s, the concept of computing clusters began to be explored using lower end or desktop computers and local area networks with systems such as NASA's Beowulf clusters[3] and UC Berkeley's NOW project.[4]

FOR MANY OF US OUR EARLIEST MODELS OF PROFESSIONALISM COME FROM OBSERVING OUR PARENTS' APPROACH TO THEIR WORK. THAT WAS THE CASE FOR ME OBSERVING MY FATHER, A SURGEON WORKING IN PUBLIC HOSPITALS IN RIO DE JANEIRO.

*Administered jointly by ACM and the IEEE Computer Society, the award is given for contributions to computer and digital systems. In 2020, my award was given for pioneering the design of warehouse-scale computing and driving it from concept to industry.

0272-1732 © 2021 IEEE
Digital Object Identifier 10.1109/MM.2021.3055379
Date of current version 26 March 2021.

When I arrived at Google, in 2001, I found a company of brilliant programmers that was short on cash but not on confidence as they had already committed to a strategy of systems built from inexpensive desktop-class components. Cheap might be a fairer characterization of those early systems than inexpensive. The first generation of those computer racks, tenderly nicknamed "corkboards" consisted of desktop motherboards loosely resting on sheets of cork that isolated the printed circuit boards from the metal tray, with disk drives themselves loosely resting on top of DIMMs.

Despite my hardware background,[†] I had joined Google to try to become a software engineer. In my early years, I was not involved in building computers but instead I worked developing our index searching software and related software infrastructure components such as load balancers and remote procedure call libraries. Three years later, Urs Hölzle asked me to build a hardware team capable not only of building sound server-class systems but to invent new technologies in the datacenter space. The years I had spent in software development turned out to be extremely useful in this new role since my first-hand understanding of Google's software stack was essential to architecting the machinery needed to run it. We published some of those early insights into the architectural requirements for Google-class workloads in an *IEEE Micro* paper in 2003.[6]

OUR TEAM'S LACK OF EXPERIENCE IN DATACENTER DESIGN MAY HAVE BEEN AN ASSET AS WE SET OUT TO QUESTION NEARLY EVERY ASPECT OF HOW THESE FACILITIES WERE DESIGNED

In our earliest days as a hardware team we focused primarily on designing servers and datacenter networking, but quickly realized that we would need to design the datacenters themselves. Up until that point internet companies deployed computing machinery in third-party colocation facilities (businesses that provisioned space, power, cooling, and internet connectivity for large scale computing gear), and Google was no exception. As the scale of our deployments grew, the minimum footprint required for a Google cluster was beginning to be larger than the total size of existing co-location facilities, so we had to build our own facilities in order to continue to grow our services.

At that point, it became evident to us how much room for improvement there was in the design of datacenters. As a third-party hosting business, datacenters were put together by groups of disjoint engineering crafts that knew little of each other's disciplines; civil engineers built the building, mechanical engineers provisioned cooling, electrical engineers distributed power, hardware designers built servers, software engineers wrote internet services. The lack of cross-disciplinary coordination resulted in facilities that were both expensive and incredibly energy inefficient. Our team's lack of experience in datacenter design may have been an asset as we set out to question nearly every aspect of how these facilities were designed. Perhaps most importantly we had the chance to look at the entire system design, from cooling towers to compilers, and that perspective quickly revealed significant opportunities for improvement.

Speed of deployment was also a critical factor in those days as we were often running dangerously close to exhausting our computing capacity as our traffic grew, so our initial approach was to prefabricate ready-to-deploy computer rooms inside forty foot shipping containers. Containers gave us a datacenter floor where we could isolate the hot (exhaust) plenum from the cold aisle and shortened the total distance the air needed to be moved; both were factors that improved cooling efficiency. All that the container needed to function was power, cold water and networking, and we had a 1200-server machine room ready to deploy.

That original container-based deployment also introduced other innovations that led to cost, performance, and energy efficiency improvements. Here are some of the most notable ones:

- *Higher temperature air cooling:* We determined through field experiments that contrary to common wisdom the electronic components believed to be most affected by air temperature were still quite reliable at reasonably high temperatures (think 70F instead of 60F).[8] This made it possible to run many facilities using evaporative cooling and improved cooling efficiency.
- *Distributed uninterruptible power supplies (UPS)*: Typical datacenters were built with a UPS room (a room full of batteries) in order to store enough energy to ride electrical grid glitches. As such ac voltage was rectified to power the UPS and then inverted to distribute to the machine room only then to be rectified again by per-server power supplies, incurring losses at each transformation

[†]My Ph.D. and the earlier phase of my career had been in computer architecture, particularly in microprocessor and memory system design.

FIGURE 1. A Google warehouse-scale computer in Belgium.

step. We instead eliminated the UPS room and introduced per tray (and later per rack) batteries. That way power entering the building only needed to be rectified once per machine.

- *Single-voltage rail power supplies*: Every server used to be outfitted with a power supply that converted ac voltage into a number of dc voltage rails (±12 V, ±5 V ±3.3 V, etc.) based on old standards for electronic components. By 2005, most electronic components did not use any of the standard dc rails so yet another set of dc/dc conversions needed to happen onboard. The allocation of power among multiple rails also lowered power supply efficiency sometimes below 70%. We introduced a single-rail power supply that reached 90% efficiency and created on-board only the voltages actually used by components.
- *1000-port GigE Ethernet switch:* Datacenter networking bandwidth was beginning to become a bottleneck for many warehouse-scale applications, but enterprise-grade switches were not only very expensive but also lacked offerings for large numbers of high bandwidth endpoints. Using a collection of inexpensive edge switches configured as a multistage network, our team created the first of a family of distributed datacenter networking products (codenamed Firehose) that could deliver a gigabit of nonoversubscribed bandwidth to up to a thousand servers.

Although our adventure with shipping containers lasted only that one generation and soon after we found ways to obtain the same efficiencies with more traditional building shells, the innovations from that first program have continued to evolve into industry-leading solutions over generations of warehouse-scale machines. Figure 1 shows a birds-eye view of a modern Warehouse-scale computer.

MY JOURNEY

I knew I wanted to be an electrical engineer when I was 8 years old and got to help my grandfather work on his HAM radio equipment. Putting aside the fact that eight-year-olds should not be making career choices, I find it difficult to question that decision to this date. Although I had always been a good student, I struggled a bit during my Ph.D. and graduated late. I did have a few things going for me: an ability to focus, stamina for hard work, and a lot of luck. As an example, after a 24-year drought the Brazilian men's national soccer team chose to win a World Cup, during my hardest year in graduate school, delivering a degree of joy that was badly needed to get me to the finish line. Less than a year after that World Cup I was working in my grad student office on a

Saturday afternoon when I got a call from Norm Jouppi inviting me to interview for a research job at Digital Equipment's Western Research Lab (WRL). At the time Norm was already one of the most highly respected computer architects in the world and perhaps nothing in my career since has compared to the feeling I had that day—Norm Jouppi knew who I was!

I KNEW I WANTED TO BE AN ELECTRICAL ENGINEER WHEN I WAS 8 YEARS OLD AND GOT TO HELP MY GRANDFATHER WORK ON HIS HAM RADIO EQUIPMENT. PUTTING ASIDE THE FACT THAT EIGHT-YEAR- OLDS SHOULD NOT BE MAKING CAREER CHOICES, I FIND IT DIFFICULT TO QUESTION THAT DECISION TO THIS DATE.

I joined DEC WRL and had the chance to learn from top researchers like Kourosh Gharachorloo and collaborate with leading computer architects such as Sarita Adve, Susan Eggers, Mateo Valero, and Josep Lariba-Pey. During that time, I also met Mark Hill who would become a friend and a mentor. Later, at Google I would also have the chance to coauthor papers with other leading figures in our field such as Tom Wenisch, Wolf Weber, David Patterson, and Christos Kozyrakis.

Perhaps nothing summarizes the impact that friends and luck can have in your life more than the story of how I came to join Google. As I was trying to make a decision between two options, Jeff Dean asked me whether the other company I was considering had also served me crème brûlée during my interviews. I thanked Jeff and accepted the Google offer the very next morning.

The brilliance and generosity of countless people at Google have been essential to the work that led to this award, but I must highlight three here: Urs Hölzle who has been a close collaborator and possibly the single person most to blame for Google's overall systems infrastructure successes; Bart Sano who managed the Platforms team that built out the infrastructure we have today (I was the technical lead for for Bart's team for many years); and Partha Ranganathan who is our computing technical lead today and is taking Google's architectural innovation into the future.

One part of my career I have no hesitation to brag about is the quality of the students I have had a chance to host as interns at DEC and Google. They were (to date) Partha Rahganathan, Rob Stets, Jack Lo, Sujay Parekh, Ed Bugnion, Alex Ramirez, Gautham Thambidorai, Karthik Sankaranarayanan, David Meisner, and David Lo. We worked together on many fun projects and I hope for more in the future. Although my dad is no longer with us, I am also fortunate to count on the love and support of my family. My mom Cecilia, my godmother Margarida, my siblings Paula, Tina, and Carlos and their families, and my wife Catherine Warner who is the award life gives me every single day.

PERHAPS NOTHING SUMMARIZES THE IMPACT THAT FRIENDS AND LUCK CAN HAVE IN YOUR LIFE MORE THAN THE STORY OF HOW I CAME TO JOIN GOOGLE. AS I WAS TRYING TO MAKE A DECISION BETWEEN TWO OPTIONS, JEFF DEAN ASKED ME WHETHER THE OTHER COMPANY I WAS CONSIDERING HAD ALSO SERVED ME CRÈME BRÛLÉE DURING MY INTERVIEWS. I THANKED JEFF AND ACCEPTED THE GOOGLE OFFER THE VERY NEXT MORNING.

THREE LESSONS

I will finish this essay by sharing with you three lessons I have learned in this first half of my career, in the hope that they may be useful to engineers who are at an earlier stage in their journey.

Consider the Winding Road

As an engineer you stand on a foundation of knowledge that enables you to branch into many different kinds of work. Although there is always risk when you take on something new, the upside of being adventurous with your career can be amazingly rewarding. I for one never let my complete ignorance about a new field stop me from giving it a go.

As a result, I have worked in areas ranging from chip design to datacenter design; from writing software for web search to witnessing my team launch satellites into space; from writing software for Google Scholar to using ML to automatically update Google Maps; from research in compiler optimizations to deploying exposure notification technology to curb the spread of Covid-19.[8]

It seems a bit crazy, but not going in a straight line has worked out really well for me and resulted in a rich

set of professional experiences. Whatever the outcome, you will be inoculated from boredom.

Develop Respect for the Obvious

The surest way to waste a career is to work on unimportant things. I have found that big, important problems have one feature in common: they tend to be straightforward to grasp even if they are hard to solve. Those problems stare you right in the face. They are obvious and they deserve your attention.

Let me give you some examples by listing some of my more well-cited papers next to the formulation of the problems address:

Publication	Problem addressed
ISCA'98: "Memory System Characterization of Commercial Workloads"[10] with Kourosh Gharachorloo and Edouard Bugnion	"High-end microprocessors are being sold to run commercial workloads, so why are we designing them for number crunching?"
ISCA'00: "Piranha: A Scalable Architecture Based on Single-Chip Multiprocessing"[5] with Kourosh Gharachorloo, Robert McNamara, Andreas Nowatzyk, Shaz Qadeer, Barton Sano, Scott Smith, Robert Stets, and Ben Verghese	"Thread-level parallelism is easy. Instruction level parallelism is hard. Should we bet on thread-level parallelism then?"
CACM '17: "The Attack of the Killer Microsecond"[11] with Mike Marty, Dave Patterson, and Partha Ranganathan	"If datacenter-wide events run at microsecond speeds, why do we only optimize for millisecond and nanosecond latencies?"
CACM '13: "The Tail at Scale"[12] with Jeff Dean	"Large scale services should be resilient to performance hiccups in any of their subcomponents"
IEEE Computer '07: "A Case for Energy-proportional Computing"[13] with Urs Hölzle	"Shouldn't servers use little energy when they are doing little work?"

If it takes you much more than a couple of sentences to explain the problem you are trying to solve, you should seriously consider the possibility of it not being that important to be solved.

Even Successes Have a "Sell-By" Date

Some of the most intellectually stimulating moments in my career have come about when I was forced to revisit my position on technical matters that I had invested significant time and effort on, especially when the original position had a track record of success. I will present just one illustrative example.

I JOINED GOOGLE AFTER A FAILED MULTIYEAR CHIP DESIGN PROJECT AND AS SUCH I IMMEDIATELY EMBRACED GOOGLE'S DESIGN PHILOSOPHY OF STAYING AWAY FROM SILICON DESIGN OURSELVES.

I joined Google after a failed multiyear chip design project and as such I immediately embraced Google's design philosophy of staying away from silicon design ourselves. Later as the technical lead of Google's datacenter infrastructure, I consistently avoided using exotic or specialized silicon even when they could demonstrate performance of efficiency improvements for some workloads, since betting on the low cost base of general purpose components consistently proved to be the winning choice. Year after year, betting on general purpose solutions proved successful.

Then, deep learning acceleration for large ML models arose as the first opportunity in my career to build specialized components that would have both broad applicability and dramatic efficiency advantages when compared to general purpose designs. Our estimates indicated that large fractions of Google's emerging AI workloads could be executed in these specialized accelerators with as much as a 40× cost/efficiency advantage over general purpose computing.

That was a time to ignore the past successes of betting on general purpose off-the-shelf components and invest heavily on the design and deployment of our own silicon to accelerate ML workloads. Coming full circle, this meant that it was now my time to call Norm Jouppi and ask him to join us to become the lead architect for what was to become our TPU accelerators program.

CONCLUDING

Before the onset of the current pandemic, some of us may have underappreciated how important computing technology and cloud-based services have become to our society. In this last year, these technologies have allowed many of us to continue to work, to connect with loved ones, and to support each other. I am grateful to all of those at Google and everywhere in our industry who have built such essential technologies, and I am inspired to be working in a field with still so much potential to improve people's lives.

REFERENCES

1. W. H. Davenport Adams, "Imitators and plagiarists," *The Gentleman's Magazine*, Jan. 1892
2. N. P. Kronenberg, H. M. Levy, and W. D. Strecker, "VAXcluster: A closely-coupled distributed system," *ACM Trans. Comput. Syst.*, vol. 4, May 1986, Art. no. 130. [Online]. Available: https://doi.org/10.1145/214419.214421
3. T. Sterling, D. Becker, M. Warren, T. Cwik, J. Salmon, and B. Nitzberg, "An assessment of Beowulf class computing for NASA requirements: Initial findings from the first NASA workshop on Beowulf-class clustered computing," in *Proc. IEEE Aerosp. Conf.*, 1998, pp. 367–381.
4. T. E. Anderson, D. E. Culler, and D. Patterson, "A case for NOW (Networks of Workstations)," *IEEE Micro*, vol. 15, no. 1, pp. 54–64, Feb. 1995.
5. L. A. Barroso *et al.*, "Piranha: A scalable architecture based on single-chip multiprocessing," in *Proc. 27th Annu. Int. Symp. Comput. Archit.*, 2000, pp. 282–293.
6. L. A. Barroso, J. Dean, and U. Holzle, "Web search for a planet: The Google cluster architecture," *IEEE Micro*, vol. 23, no. 2, pp. 22–28, Mar./Apr. 2003.
7. A. Singh *et al.*, "Jupiter rising: A decade of Clos topologies and centralized control in Google's datacenter network," *SIGCOMM Comput. Commun. Rev.*, vol. 45, no. 4, pp. 183–197, Oct. 2015.
8. E. Pinheiro, W. Weber, and L. Barroso, "Failure trends in a large disk drive population," in *Proc. 5th USENIX Conf. File Storage Technol.*, Feb. 2007, pp. 17–29.
9. Google & Apple Exposure Notification technology. 2020. [Online]. Available: g.co/ENS
10. L. A. Barroso, K. Gharachorloo, and E. Bugnion, "Memory system characterization of commercial workloads," *SIGARCH Comput. Archit. News*, vol. 26, no. 3, pp. 3–14, Jun. 1998.
11. L. Barroso, M. Marty, D. Patterson, and P. Ranganathan, "Attack of the killer microseconds," *Commun. ACM*, vol. 60, no. 4, pp. 48–54, Apr. 2017.
12. J. Dean and L. A. Barroso, "The tail at scale," *Commun. ACM*, vol. 56, no. 2, pp. 74–80, Feb. 2013.
13. L. A. Barroso and U. Hölzle, "The case for energy-proportional computing," *Computer*, vol. 40, no. 12, pp. 33–37, Dec. 2007.

LUIZ ANDRÉ BARROSO is a Google Fellow and a former VP of Engineering at Google. His technical interests include machine learning infrastructure, privacy, and the design and programming of warehouse-scale computers. He has published several technical papers and has co-authored the book *The Datacenter as a Computer*, now in its 3rd edition. He is a Fellow of the ACM and the AAAS and he is a member of the National Academy of Engineering. Barroso received a B.S. and an M.S. in electrical engineering from the Pontifícia Universidade Católica of Rio de Janeiro, Rio de Janeiro, Brazil, and a Ph.D. in computer engineering from the University of Southern California, Los Angeles, CA, USA. He is the recipient of the 2020 Eckert-Mauchly award. Contact him at luiz@barroso.org.