HPC TECHNOLOGY SPECIAL

CIOReview

The Navigator for Enterprise Solutions

MARCH 2-2015 CIOREVIEW.COM

Company of the Month

Gary Gauba, Chairman & CEO, Cognilytics

Entrepreneur of the Month

Richard E. Malinowski Founder & President, REMTCS

Nadia Short, VP & GM, Cyber Systems, General Dynamics Mission Systems

General Dynamics Mission Systems

Delivering Real-time Intelligence through HPC

$15 US

CIO REVIEW
44790, S Grimmer Blvd.
#202, Fremont, CA-94538

CXO INSIGHT

Scale Up and Out:
The Changing Face of High Performance Computing

By Jeffrey M. Birnbaum, Co-Founder & CEO, 60East Technologies, Inc.

If you were designing and building the Google infrastructure today, what would you build?

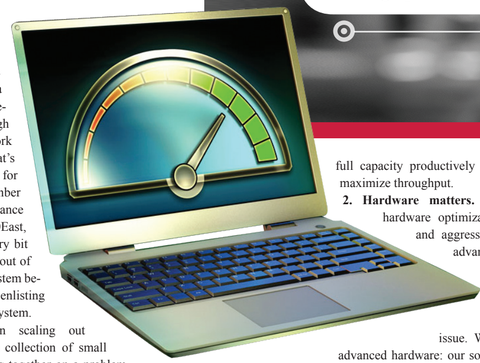
It's an interesting question, and one that doesn't have a simple answer. The world has changed since the early days of Google. In 1998, a high performance commodity server was a dual-processor system with each processor on a dedicated socket, running at speeds of 200-300MHz. Those systems had 256 –512MB of RAM and a few hundred GB of disk spread across multiple 9GB hard drives running at 7200RPM. 100Mbps Ethernet was the high-end networking protocol. By today's standards, these systems are memory-hobbled; networks constrained, storage deprived, and suffer from very limited ability to parallelize on a single system. Google solved their problem by building a bigger system composed of these small systems. There wasn't enough bandwidth on a single system, so they used more systems. A single system didn't have enough processor power, and could only run a few concurrent processes, so Google ran tasks on more systems. To get capacity, Google added complexity by distributing the work across their fabric, and added cost in the need for more infrastructure and coordinating systems. Google also designed the system to return an answer quickly, even if the answer was based on data minutes, hours, or days out of date. Google's platform is a monumental engineering achievement, but the question remains: if you were going to build the platform

Today, would you make the same choices?

Today, you can order a single high-end server off the shelf that has more processing power, network bandwidth, memory, and disk capacity than a full rack of servers had in 1998. A high-end server these days has 36 cores spread across 2 sockets with each core running at 2GHz or faster, a terabyte of memory, and dozens of terabytes of storage. 40 gigabit Ethernet is common, with 100 gigabit Ethernet on the horizon.

The problems are also tougher. Applications need to do their work faster than ever before. At 60East, we call this the drive to "real-time". Our customers measure end-to-end performance in microseconds while keeping millions of live records up to date. With our customers, we live high-performance computing in the real world. Like Google, though not yet at the same scale, our customers process message streams at ever increasing pace and volume. Unlike Google, our customers can't tolerate stale data. Businesses are at stake if their systems can't keep up. The 60East Advanced Message Processing System (AMPS) processes billions of messages daily, in some of the world's largest financial institutions. We've learned what works, and what doesn't, through experience.

With today's abundance of capacity, the problem isn't finding enough processor power or storage to handle the data. Today, performance is about making sure we fully use the power in our existing systems. Modern servers can analyze data and produce results fast enough to fill the network capacity, and that's enough power for an amazing number of high performance problems. At 60East, we squeeze every bit of performance out of a single large system before we turn to enlisting more than one system.



Rather than scaling out first, creating a collection of small systems working together on a problem, we get better performance by scaling up first. Only after we've reached capacity on a large system do we scale out. This approach takes full advantage of modern systems, and reduces the complexity and overhead required to keep different systems working together.

To produce world class performance in today's High Performance Computing world, we've developed these principles:

1. Maximize concurrency, not just parallelism. It's commonplace to divide a task across processor cores to take advantage of parallelism. We also look for ways to run different tasks at the same time, and try to use every bit of capacity the CPU has. Idle CPU cycles are a wasted opportunity. Our goal is to use a system's full capacity productively to minimize latency and maximize throughput.

2. Hardware matters. We take advantage of hardware optimizations wherever possible, and aggressively build for the most advanced systems we can get our hands on. In this industry, Hardware that is cutting edge today will soon be standard issue. We also engineer for less advanced hardware: our software scales both up and down and adapts to take advantage of the optimizations available. Abstraction from the hardware is great for ease of development, and we use high-level abstraction when we can. For peak performance, though, it all comes back to the machine.

3. Memory is slow and massive multicore changes the game. We have more raw processing power than ever before, but memory speeds haven't kept up. Memory locality matters: in our testing, we've seen as much as a 10x performance bump just from making sure that code runs on a processor near the memory it uses.

4. Sweat the details, again and again. It's easy to find the big hotspots in performance. We love finding an efficient algorithm or the perfect data structure. It's harder to find the tiny slowdowns that accumulate throughout the system. The total effect of those tiny slowdowns can be even larger than the big hotspots. There's only one way to keep things fast, and that's to pay attention to every line of code and every data structure.

5. Keep it simple. Fast code only matters when the code does something important. Don't do unnecessary work or add unnecessary features. If code is too complicated to review confidently for performance or rewrite to go faster, then it's become too complicated. If the code isn't important enough to go under the performance microscope, then what it does isn't important enough to be part of your system. Prefer simple solutions, and resist the urge to add convenience at the expense of performance.

6. The conventional wisdom isn't always wise. Conventional approaches were developed for the systems available at the time, and they worked well for that environment. In today's world, those approaches can be slow and inefficient. For example, traditional databases spend time managing indexes for faster retrieval, yet the overhead that comes with index maintenance often outweighs the performance advantages on today's systems. Using a divide and conquer approach that avoids locks often runs faster, and we use that approach in AMPS. Conventional wisdom may say this is the wrong approach, but measurement says that the "wrong approach" is the better solution.

7. Everything works together. Applying just one or two of these principles won't give you high performance consistency. To create high performance software, and maintain that performance as your product evolves, it's important to apply these principles to every update and every change.

It may sound simple, but the work involved isn't trivial. There are no shortcuts, but the results are worth it. AMPS demonstrate these principles in action, and our customers reap the rewards.

The ongoing changes in the computing landscape give you the opportunity to change what you're doing and create systems that deliver more scale and performance, at a lower cost per transaction, than ever before. It's an exciting time in high performance computing, and 60East is proud to be a part of it.

Are you ready for the future? CR

> **Modern servers can analyze data and produce results fast enough to fill the network capacity, and that's enough power for an amazing number of high performance problems**

Jeffrey M. Birnbaum