THE VOLATILE FUTURE OF STORAGE

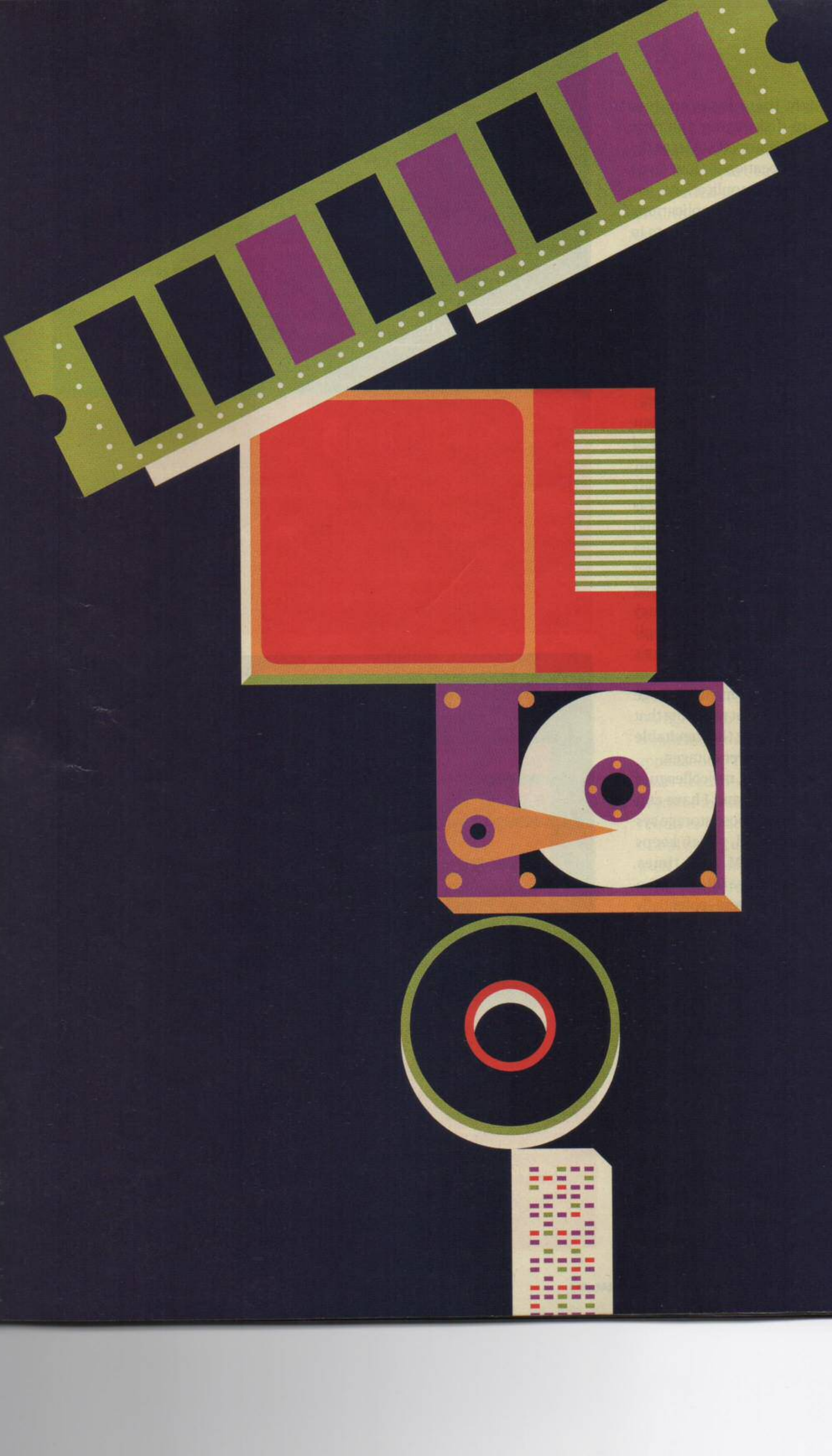
Speedy DRAM could replace hard disks for cloud computing

by **JOHN OUSTERHOUT**

WHEN IT COMES TO COMPUTER STORAGE, the magnetic disk has been top dog for almost half a century. The first commercial disks appeared in 1956, and by the early 1970s their cost and capacity had improved to the point where they began to replace magnetic tape as the primary storage medium for computers. By the end of that decade, tapes had been relegated mostly to a backup role. Since then, disk technology has improved at an exponential rate, just like integrated circuits. Nowadays, a typical drive holds 20,000 times as much data as it did in 1985, and on a per-byte basis, disks cost one-millionth of what they did then.

No wonder hard disks are so pervasive. This is also why today's popular forms of computer storage, such as file systems and relational databases, were designed with disks in mind. Indeed, until recently any information kept on a computer for more than a few seconds probably ended up on disk. ● But the hard disk's reign may be coming to an end. The most obvious challenger is flash memory, which is faster, more compact, and more resistant to shock. Virtually all mobile devices, such as tablets, smartphones, and watches, already use flash instead of disk. Flash memory is also displacing hard drives in laptops and, increasingly, in large-scale applications running in data centers, where its speed is a significant advantage.

ILLUSTRATION BY Greg Mably

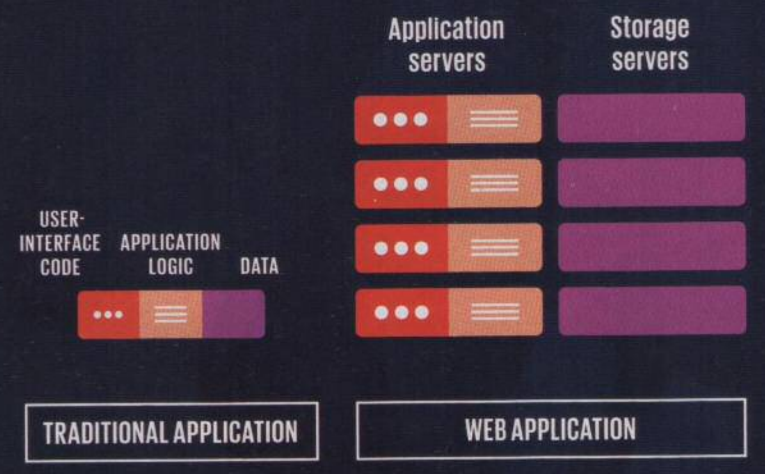


Now though, there is yet another alternative to disk: using dynamic random-access memory (DRAM) as the primary storage location for long-lived data. More and more applications, particularly large-scale Web applications, are keeping most or all of their data in DRAM. For example, all of the popular Web search engines, including Google, service people's queries entirely from DRAM. Also, Facebook keeps most of its social-network data in DRAM. And IBM's Watson artificial-intelligence system kept all of its data in DRAM when it won the "Jeopardy!" challenge a few years ago.

On the surface, this seems ridiculous. After all, DRAM was intended to hold information temporarily during active computations. Although it is about 1,000 times as fast as flash, it is also 100 times as expensive as disk, and it is volatile, which means that the data it holds will disappear if the computer loses power. Nevertheless, I believe that DRAM could soon become the primary storage medium for large-scale applications running in data centers. Here's why: If DRAM is backed up on disk or flash, users can enjoy the medium's huge speed advantage without worrying that data will be lost during the inevitable server crashes and power outages.

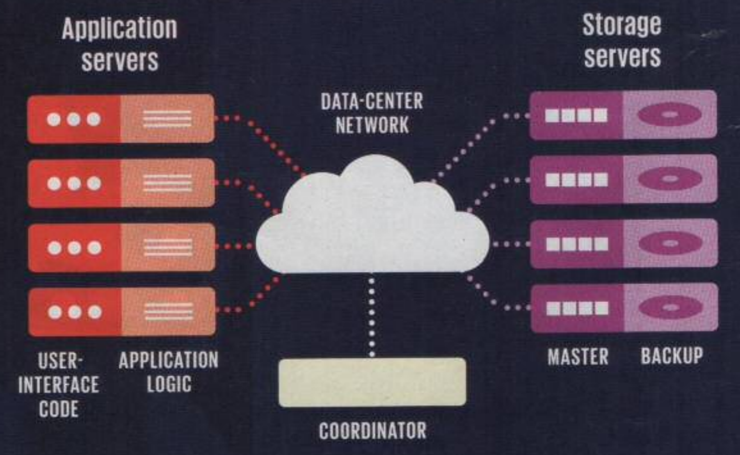
As a research project, my colleagues at Stanford University and I have constructed a general-purpose storage system we call RAMCloud, which keeps all of its data in DRAM at all times. RAMCloud aggregates the DRAM memories of a collection of servers—potentially hundreds or thousands in a typical data center. To work around DRAM's volatility, RAMCloud stores copies of data on disk or in flash memory, and it automatically recovers data from those backups after a server crashes. Our hope in the RAMCloud project is to make it as easy for developers to use DRAM-based storage as it is for them to use disk.

OUR MAIN MOTIVATION FOR MOVING FROM disk to DRAM comes from the evolution of disk technology. Although the storage capacity of disks has mushroomed over the years, their access speed has not improved as much. The reason for



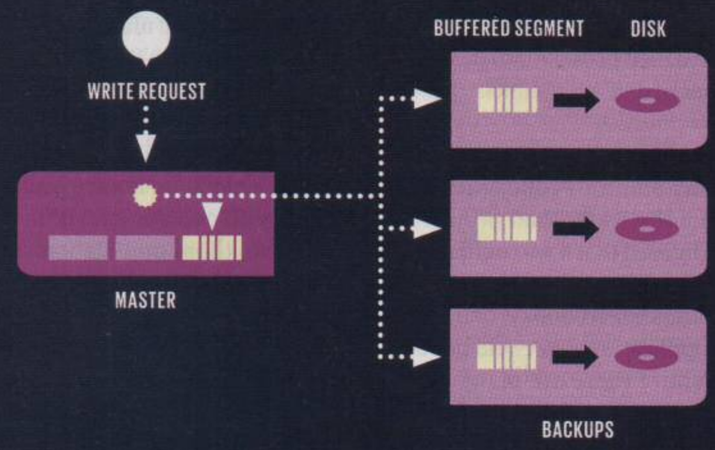
WEB-SCALE APPS

Traditional applications manipulate data held in the memory of a single machine [left]. But large-scale Web applications require a more complicated architecture [right], with many different computers in a data center running the application logic and the server-side user-interface code. Other computers act as storage servers. They keep the bulk of the data on disk with only the most recently accessed information held in memory.



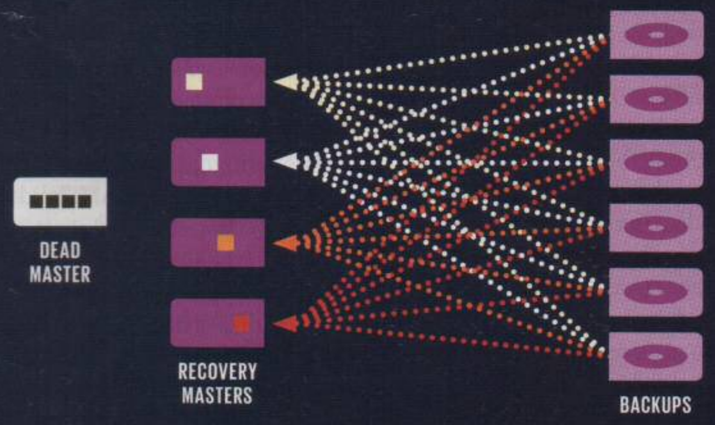
UP WITH DRAM

RAMCloud's storage servers have lots of DRAM (from 32 to 256 gigabytes on each machine)—enough in aggregate to hold all of the data. Those storage servers also contain hard disks and a small amount of battery-backed DRAM, together used for backup copies of data from other servers. A separate machine, called the coordinator, manages the communications between application servers and storage servers.



WRITE THREE DEEP

When a RAMCloud storage server receives a request to store a chunk of data, it adds the information to its master copy of the data in DRAM, which takes the form of an in-memory log. That storage server also backs up the newly recorded information on at least three other storage servers, which save redundant copies of the data, initially in battery-backed DRAM and then on disk.



CRASH 'N' BURN

When a storage server crashes, the data it held in DRAM must be reconstituted—fast. For that, several existing servers are assigned to act as *recovery masters*, restoring different portions of the lost data and taking ownership of it. Each recovery master reads from multiple disk backups, speeding the recovery process. With enough servers working together, hundreds of gigabytes can be reconstituted in just a second or two.

this is simple enough. To read or write data on a hard disk, a physical mechanism must first position the read-write head over a particular track on a spinning platter. Then the system must wait until the desired information rotates under the head. These mechanical processes have proven difficult to speed up much. With the amount being stored on a disk increasing more rapidly than the access rate, the time needed to read or write all the data on a disk has increased. Indeed, if the information you need is stored in small blocks spread randomly on the disk (which is common in many applications), the time it takes to hunt down the data becomes significant. Even if you dedicated your computer to that task, to access each block in random order could take a few years!

This problem is making disks increasingly unsuitable for storing small pieces of data that must be accessed frequently. In a sense, the hard disk is putting itself out of business. This trend has been evident for many years and explains much of the interest in flash memory, with its higher access rates.

There is a second motivation for new storage technologies that's even more compelling: the rise of large-scale Web applications. These applications sometimes need to support tens or even hundreds of millions of users. Doing that requires a radically different operational structure than what has been used traditionally.

The old approach was to load the application's code and all of its data into the main memory of a single machine. The application could then access its data at DRAM speeds (typically 50 to 100 nanoseconds), which allowed it to manipulate its data intensively. The problem was that the total throughput of the application was limited by the capabilities of that one machine.

As the Web grew in popularity during the late 1990s, it quickly became clear that the traditional application architecture could not handle the loads generated by popular websites. Over the next 10 years, designers came up with a new architecture in which thousands of machines work together in huge data centers. One of the most important attributes of this arrangement is that application code and data are separated. One set of machines (called application servers) runs the applications, while a different set of machines (storage servers) holds the data. In this architecture, data is typically stored on disk, although frequently needed data may be held for quick access (cached) in DRAM.

This new architecture has enabled the creation of applications of a scale that was unimaginable 20 years ago—Facebook and Google Search come to mind—but it has also changed the relationship between an application and its data. In the Web architecture, each read or write access requires an application server to communicate over the data center's internal network to a storage server, and the storage server may additionally have to perform a disk access. The total time to fetch a chunk of data is now 0.5 to 10 milliseconds, four to five orders of magnitude longer than what it would be with the traditional one-machine-does-everything architecture.

As a result, Web applications cannot use their data very intensively. For example, when Facebook generates a Web page for a user, it can consult only a few hundred distinct pieces of data: Anything more would take intolerably long. This restriction limits the kinds of features that Facebook can provide, and it makes life difficult for the company's developers, who must labor to pack the most information possible into the smallest number of distinct chunks.

OUR AWARENESS OF THESE PROBLEMS spurred my Stanford colleagues and me to start the RAMCloud project in 2009. We figured that if we could speed up data access, it would make an enormous difference. Facebook is one example of an application that could benefit, but we believe there are many more things that are not even attempted today because no storage system can support them.

DESIGNED FOR SWAPPING: Servers can be replaced quickly. But even so, if DRAM is used for long-term storage, crashes remain problematic. RAMCloud automates data recovery, restoring within just a couple of seconds the information lost when a server fails, so users don't notice.

To achieve that speed boost, RAMCloud keeps all data in DRAM at all times, but it also makes the data just as reliable as if it had been stored on disk. RAMCloud doesn't require any sort of exotic hardware to do so. Indeed, it's just a software package that runs on ordinary machines—RAMCloud servers, which take the place of today's storage servers.

Each RAMCloud server contains two software components: a *master* and a *backup*. The master code uses most of the server's DRAM to store RAMCloud data, and it makes that data available to the applications servers that request it. The backup code's job is to keep redundant copies of the data from other RAMCloud masters in the server's secondary storage (on disk or in flash memory).

A third software component of RAMCloud, the *coordinator*, runs on a separate machine. It manages the overall configuration of the RAMCloud cluster and coordinates recovery actions when storage servers fail. The various application servers running in the data center use a RAMCloud library package to access RAMCloud storage. The first time they do so, those application servers must contact the RAMCloud coordinator to find out which masters store which data. The application servers then cache this information locally, so subsequent RAMCloud requests can go directly to the relevant master.

While the RAMCloud servers are intended to take the place of normal storage servers, they can't entirely replicate what today's database servers do. RAMCloud's data model is a key-value store, which consists of a collection of tables, where each table can contain any number of objects. Each object consists of a variable-length key, which is unique within its table, and an associated value, which can be a blob of arbitrary data up to 1 megabyte in size. Applications read and write objects on a RAMCloud server simply by specifying a table and a key within the table. RAMCloud does not currently support all of the features of a database system, such as secondary indexes and transactions.

While some will see this feature deficit as a shortcoming, duplicating the functionality of today's database servers was not our goal. Our most important design imperative for RAMCloud was to achieve the fastest possible access times, defined as the total time, measured by a running application, to read or write small chunks of data that are stored in a RAMCloud server in the same data center. With the current version of RAMCloud, reads take about 5 microseconds (writes take about 14 μs) for 100-byte objects. That's roughly 1,000 times as fast as you could read data from a local hard disk and 10 to 20 times as fast as you could get that data from local flash memory.

When we began the RAMCloud project, access times like these were almost inconceivable. Even with data stored in DRAM, the networking infrastructure itself imposed significant delays. Each switch in the data-center network added 10 to 30 μs, and the request for data, and the supplied response, could each have

THOMAS TRUTSCHEL/GETTY IMAGES

to go through as many as five switches in a large data center. And data packets had to be processed by the operating system when entering or leaving a machine, which added roughly 60 μs. So typical round-trip times within a data center were hundreds of microseconds long.

Two improvements in networking have made RAMCloud's blazing speed possible. The first was a new kind of networking infrastructure that uses special-purpose switching chips with internal delays of less than 1 μs per switch. The second was a new generation of network-interface controllers (NICs), which have a capability called kernel bypass. This feature allows an application to communicate directly with the NIC to send and receive packets of data without involving the computer's core operating system.

The combination of fast-switching chips and kernel bypass makes ultralow-latency communication possible: less than 5 μs for requests that need to pass through only a few switches (such as in our test cluster), and around 10 μs for requests in a very large data center with 100,000 machines. With the additional improvement in networking technology we expect to see over the next 5 to 10 years, we believe that round-trip times in a large data center could be reduced to as little as 2.5 μs.

IN ADDITION TO PROVIDING FAST ACCESS, RAMCloud must also ensure that its data is stored as reliably as if it were held on disk. In particular, data must not be lost when a server crashes or the power goes down. Data centers typically lose power every few years, which can cause all the information in DRAM to be lost. As mentioned earlier, RAMCloud keeps backup copies of data on disk or in flash memory. (This scheme is analogous to the way that current disk-based storage systems keep backup copies on magnetic tape.)

But what if an individual RAMCloud server failed? To protect against such mishaps, RAMCloud keeps multiple backup copies (typically three) of every single piece of data, storing those on different servers. So when a write request arrives at a master, it updates its information in DRAM and then forwards the new data on to several backup RAMCloud servers. This need for replication explains why writing data takes longer than reading it.

In addition to these general measures, we had to solve two specific problems to make RAMCloud's backup strategy really bulletproof. The first was what to do if a server loses power before it has made backup copies. RAMCloud deals with that possibility using a small amount of nonvolatile memory on each machine. When new data is written to one machine, the backup machines associated with it collect that data temporarily in some form of fast but nonvolatile memory. That could take the form of battery-backed DRAM, for example. These backups then write the data to disk or flash in the background. Having the data in fast nonvolatile memory at the start ensures that the information can be recovered if a power failure occurs before the data is written to disk or flash.

The second problem is that a RAMCloud cluster with thousands of servers will inevitably experience frequent server crashes. RAMCloud keeps only a single copy of information in DRAM, so data that was stored on a crashed server will be unavailable until it can be reconstructed from information on the hard disks (or flash memories) of its backups. If all the crashed machine's data were held on one other disk, it would take several minutes to get it into working memory.

To avoid that long a delay, RAMCloud scatters the backup data for each master across all of the other servers in the cluster, which could amount to thousands of machines. After a crash, all of those machines work in parallel to reconstruct the lost data. With the work spread among so many computers, RAMCloud can recover from server failures in just 1 to 2 seconds—so fast that most Web users wouldn't even notice.

ALTHOUGH RAMCLOUD IS JUST A UNIVERSITY research project, our goal was to create a production-quality system that can be used for real applications. At the beginning of last year, the system reached version 1.0: The basic features were in place, and the software was mature enough for early adopters to begin experiments. Since then, several groups outside Stanford have begun experimenting with RAMCloud's open-source code, for such things as the Distributed Main Memory Database project at telecom giant Huawei.

One possible impediment is that RAMCloud does not support the relational data model that dominates information technology today. It does not provide the convenient facilities of a relational database management system, so many applications would have to be reprogrammed to use it. But that's perhaps not as big an issue as it would appear. The applications for which RAMCloud would be most advantageous haven't yet been written, simply because there is no storage system capable of supporting them.

So RAMCloud's early adopters are not likely to be those using existing enterprise applications. Instead, the first users will probably be developers who discover that today's storage systems are just not fast enough to meet their needs. These groups can design their applications around RAMCloud from the start, and they should be more willing than most to try it out. Desperate people are, after all, the friend of new technology!

If RAMCloud proves successful with its early adopters, it may not be long before it spreads into the mainstream. RAMCloud is a good match for cloud-computing data centers such as Amazon Web Services, Google Cloud, or Microsoft Windows Azure. In such an environment, the cost of a RAMCloud cluster could be amortized across many users, who could then take advantage of high-speed storage at very low cost, while retaining the ability to expand easily as their needs grow.

RAMCloud represents a new class of storage that simultaneously achieves vast capacity and low latency. It would let something like a million CPU cores within a data center work together on data sets of 1 petabyte or more, where any core can access any data item in 5 to 10 μs. That's one-thousandth of the time it typically takes now. So if you're already impressed by the speed and power of today's large-scale Web applications, fasten your seat belt and get ready for even bigger thrills in the future. ■

POST YOUR COMMENTS at http://spectrum.ieee.org/ramcloud1115