data EXPEDITION inc®

FROM SILICON TO MEMES

How We Compute Communication and the Case for a New Data Transport Model

By Seth Noble, PhD, CEO & Founder of Data Expedition, Inc.

A Foreword by Seth Noble, PhD, CEO & Founder of Data Expedition, Inc.

Given the evolution of computing over the past five decades, it is almost unthinkable that the single piece of technology which now controls nearly all global communication remains unchanged. The basic data model for TCP – the data transport protocol underneath it all – is the same as it was in 1974 and every company that depends on digital communication, whether for operations or customer engagement, is losing up to 20% of its revenue to this inefficiency. This essay discusses how we got here, the cost, and potential solutions.

If your company communicates over digital networks, you should not be waiting to think about how the efficiency of that communication affects your top and bottom lines.

Abaci to Emoji

How Automated Calculators Evolved into a Global Communications Network

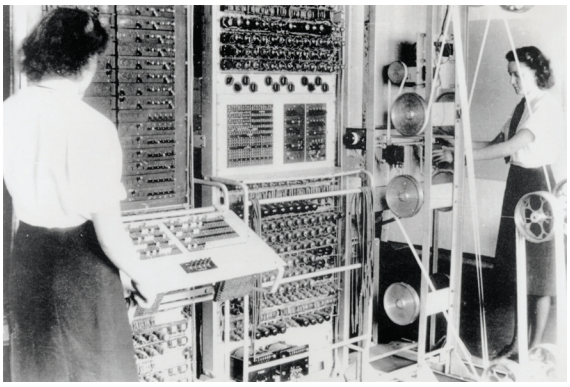
Smartphones, laptops, desktops, tablets and the many other devices that dominate our daily lives all share a common label: computer. But the most familiar use of computers is not to compute. More than anything else, computers are used to communicate. From tweets to novels, voicemails to operas, and snapshots to movies, almost all communication that travels more than a few feet is handled by computers.

Computing is involved in the sense that a CPU is executing mathematical functions in service of communication. But how we get from automated arithmetic to the mass exchange of ideas involves a lot more than just electrons following circuits. Looking at both the history of digital communication and the technical processes behind it reveals layers of complexity, which impact nearly every facet of our economy and culture.

➲ Prior to the invention of general-purpose electronic computers in the 1940s, the word "computer" was a job description.

Prior to the invention of general-purpose electronic computers in the 1940s, the word "computer" was a job description. For example, during World War II, hundreds of women operated mechanical calculators to compute artillery tables for the military. Their work led directly to the invention of programmable electronic computers for the same purpose.

As electronic computers grew faster and more flexible in their calculations, they also gained the ability to store data persistently. Magnetic tapes, drums and disks were among the devices allowing computers to keep and organize data. This organizational ability led to a major abstraction of computing: from data to information. Computers grew to allow us to rearrange our ideas, as well as record them. As mainframes gave way to microcomputers, the notion of a computer as a personal information device began to take hold.



The ability to store large amounts of organized data in a small space also made it much easier to move that data around. Whether it was a plastic case full of floppy disks or a station wagon full of tapes, it became possible to communicate immense amounts of information with relative ease. As storage media evolved, computers moved beyond numbers, text, and simple pictures to carrying high-quality voice, music and video. CDs and DVDs marked the transition of computers from mass storage to mass communication.

However, it was the interactivity of computer networks that really pushed computers into the realm of personal communication. From dial-up bulletin boards to the internet, computers have made it possible to transform our thoughts into data, transmit that data around the world, and transform it back into another person's mind in a matter of milliseconds.

Of course, mass communication and long-distance real-time interaction have been around a lot longer than computers. Whether drums, flags or just a loud voice, people have been carrying on conversations across great distances for thousands of years. Computers have added the ability to organize, store and move large amounts of data very quickly. What began as automating the movement of simple numbers has, in less than 80 years, evolved into automating the movement of abstract ideas. This is what now allows us to carry on a dozen or more different conversations at once or exchange the nuances of facial expressions and body language among dozens of faraway people.

For better or worse, this automation has changed the daily lives of most human beings.

➲ Interactivity of computer networks really pushed computers into the realm of personal communication,

while automation has changed the daily lives of most human beings.

➲ Tech fortunes rise and fall on finding novel ways to represent and organize our ideas, but the mechanism underneath it all has remained largely unchanged since 1974.

It would be natural to assume that this rapid change in how computers are used has been accompanied by an equally rapid change in how they work. For many aspects of hardware and software, this is true. But there are some aspects of computing technology that have lagged decades behind.

Conceptually, platforms like Twitter, Instagram, Skype, FaceTime or venerable email all do basically the same thing: they encode some aspect of our thoughts, communicate that data, and then decode it to be consumed by other people. What distinguishes each of them is how they perform that encoding and decoding – which aspects of ourselves they record, as well as how they organize and represent the information to others. Those technologies continue to evolve, and tech fortunes rise and fall on finding novel ways to represent and organize our ideas.

But underneath all those familiar platforms, the same software is used to manage the actual movement of every type of data. In fact, nearly all modern communication uses the Transmission Control Protocol (TCP) to perform the computing involved in figuring out how to move those bits from one place to another. Even more surprising is that this mechanism has remained largely unchanged since 1974. Imagine a car built today with a carburetor from 1974 controlling its fuel flow. Or imagine trying to build an electric car with a carburetor as a required component.

To be clear, TCP has had many tweaks and adjustments, but the data model of a generic, bi-directional byte pipe with a single 32-bit number controlling security, flow control and error recovery remains the same as it was nearly 50 years ago. Some of TCP's legacy can be attributed to its brilliantly simple and general-purpose design. But considering the vital role played by digital communication in our lives, culture and economy, a deeper look at exactly how this lynchpin of modern life works and why it perseveres is more than warranted.

The Mechanisms of Digital Communication

For more than a century, electronic communications such as telegraphs and telephones consisted of an analog circuit of physical wires connecting devices. The circuit might be thousands of miles long and pass through many junctions and repeaters, but an electrical signal at one end propagated all the way to the other. The public switched telephone network (PSTN) allowed such connections to be created on demand, but the communication was still basically a wired circuit between two points.

Digital communications provide an abstraction away from the vagaries of electrical signal propagation. But early digital devices such as Teletype terminals and modems came to rely on the PSTN as a convenient way to establish a circuit between distant systems. This concept of circuit-switched networks carried over into technologies like Asynchronous Transfer Mode (ATM), which was once touted as the future of digital communications. Computers were expected to call one another, exchange information, and then release the circuit.

Packet-switched networks brought a very different approach. Messages between systems are broken into small packets of bytes. Much like a letter through the postal system, each packet carries its destination address and return address. This allows many packets to be sent to, or received from, many different destinations without having to establish a dedicated circuit each time.

Packet switching added tremendous flexibility to digital communications but also introduced several new problems. First, nearly all computer software still communicated point-to-point in streams of bytes, not small packets. Second, packet-switched networks such as ARPANET (and its descendant, the internet) are "best effort," meaning there is no guarantee any particular packet will make it all the way to its destination. Third, a device might know the speed of its immediate connection, but the capabilities of the network beyond are completely unknown and could change at any time. Send data too fast, and packets will simply disappear.

➲ Circuit-switched networks carried over into networks... computers were expected to call one another, exchange information, then release the circuit.

Vinton Cerf

Robert Kahn

1974: Vinton Cerf and Robert Kahn came up with an elegant solution to meet the needs of the day, **Transmission Control Protocol (TCP).**

In 1974, Vinton Cerf and Robert Kahn came up with an elegant solution to meet the needs of the day. Their Transmission Control Protocol (TCP) – originally called the Transmission Control Program – solved all three packet switching problems by creating a virtual circuit on top of the packet-switched network. TCP automatically breaks up a stream of bytes into packets and reassembles those packets into the original order at the other side. It detects and repeats lost packets and uses that loss as a signal to temporarily slow down to avoid more loss. This software system of reliable, congestion-controlled byte pipes allowed distant software to communicate as if it had a virtual circuit, without having to worry about the details of packets.

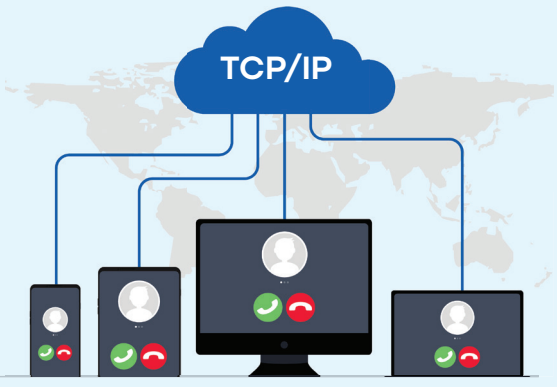
Today, practically all digital communication networks are based on the Internet Protocol (IP) and nearly all applications use TCP. While some implementation details have changed, modern TCP still uses the exact same full-duplex, virtual-circuit model it did nearly 50 years ago. All of those popular applications, all of the ideas they carry, and all the innovations they create still pass through this one technology.

While that is a great testament to TCP's design, a lot has changed since then. Almost every aspect of modern networking has not only grown by a factor of millions but has also become millions of times more varied. Speeds range from tens of kilobits to tens of gigabits per second, latency (the time for a packet to cross the network) ranges from microseconds to tens of seconds, and millions of simultaneous data flows compete for bandwidth over backbone paths.

In the face of such change, TCP has shown its age. Enterprise users, whose livelihood depends on moving large amounts of data quickly, see that data sent by TCP often fails to reach anywhere near their network hardware capacity. The result is consumers see slow-loading web pages, hung applications, and delayed messages. Buffering and dropped connections are a part of internet life. Ever faster hardware speeds have only made the problem more obvious: TCP was never intended to scale this far.

With nearly every aspect of our modern culture and economy dependent upon digital communication, even seemingly small inefficiencies can have profound effects.

➲ Today, practically all digital communication networks are based on the Internet Protocol (IP) and nearly all applications use TCP that still uses the exact same full-duplex, virtual-circuit model it did nearly 50 years ago.

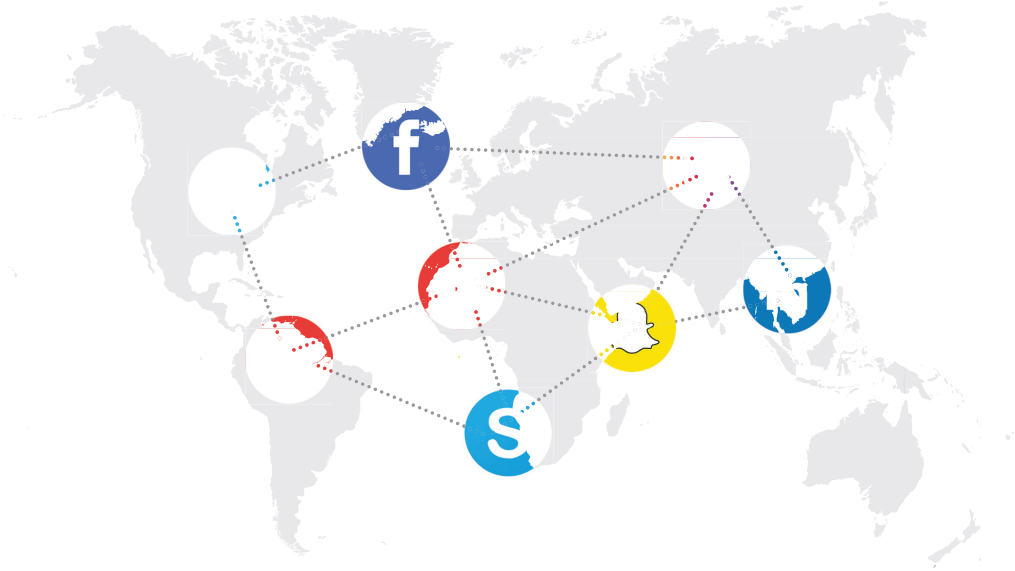


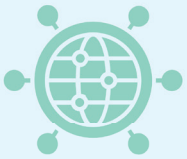
The Cost of Communicating with a Protocol from 1974

Consider a 10-petabyte video vault that needs to be moved into the cloud for the launch of a streaming service. (Data sets of that size are a lot more common than most people realize, and so is the need to move them around.) Bringing a 10 gigabit-per-second data pipe to the vault should move the data in about three-and-a-half months. But with TCP often achieving less than a quarter utilization across long distance paths, the project could find itself delayed by nearly a year. (This example is grossly simplified, but the scaling issues hold.) **Imagine the cost of any enterprise project being multiplied by a factor of four. Imagine the cost of all enterprise projects being delayed.**

To say that data transport inefficiency is costing businesses hundreds of billions of dollars per year is very likely understating the problem. But the effects of inefficient network communication reach far beyond a few big companies or niche industries. With our entire economy and culture dependent on real-time communication, even small inefficiencies have huge effects.

Former Google VP Marissa Mayer famously observed in 2006 that just a half-second delay in results dropped Google search traffic by 20 percent. Google's websites generated nearly 100 billion dollars in 2018, so half a second of delay translates to nearly 20 billion dollars. Amazon (per data from former Amazon engineer Greg Linden), Walmart and others have identified similar costs in the past (though most now view this as an area of proprietary optimization and no longer share revenue impact). Multiply those effects across every user of Google, Facebook, YouTube, Twitter, Instagram, Snapchat and all the other communication platforms in use today and the cost of TCP grows to impact our entire global economy.





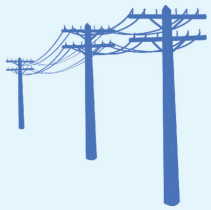
With our entire economy and culture dependent on real-time communication, even small inefficiencies have huge effects.

Google

Just a half-second delay in results dropped **Google** search traffic by

20%

Former Google VP Marissa Mayer 2006

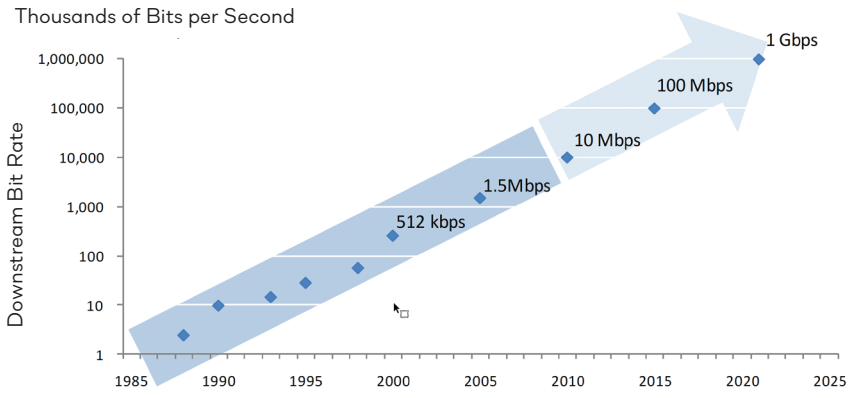


➲ Inspired by analog telephone lines, TCP assumes data must be delivered in order, data must be able to flow in both directions at the same time, and that network congestion will be rare. Modern applications and networks have left those assumptions behind.

The key to solving this problem is understanding where TCP's evolution stalled. Inspired by analog telephone lines, its full-duplex, virtual-circuit model assumes data must be delivered in order, data must be able to flow in both directions at the same time, and that network congestion will be rare. However, the furious innovations of modern applications and the astounding scale of modern networks have left those assumptions behind.

Ensuring that every application receives all data in order requires that TCP store the data in a buffer. The size of the buffer limits the amount of data that can be in flight on the network, which in turn limits the speed at which data can be moved. (Search for "bandwidth delay product" for more about the relationship between buffer size, latency, and speed.) For bulk data and interactive user data, the kinds most sensitive to delays, such ordering is rarely necessary. Consider the transfer of webpage image files: While some formats can be displayed progressively, none require it and vanishingly few benefit from it. What matters to users is when the image is ready to view in its entirety. By forcing ordered delivery on all data transfers, TCP places an unnecessary burden on finite resources and unnecessary limits on speed and latency tolerance.

A full-duplex (bi-directional) byte pipe makes for a very general communication model, but the majority of network communications consists of one-way bursts. Loading a webpage or transferring a file involves sending a few small requests, sometimes to many different servers, after which the data flows in one direction. When the size of each data flow is small, the time it takes to create and shutdown down each TCP session can be many times what it takes to transfer the actual data. When the data is large, maintaining an unused backchannel and the ability to start and stop data flow at the source forces compromises in congestion control and error recovery that further limit scaling.



Bandwidth Grows by 10x every 5 years

Source: NBN Co, Alcatel-Lucent (refer Exhibit 9.21)

Today, everyone is familiar with network congestion and its resulting slowdowns. But TCP was designed around the idea that it would instead be the receiving computer that would have trouble keeping up with the network. As a result, network congestion causes much worse problems than merely slowing down TCP. Because of the ordering requirement and the finite buffer, TCP must stop for holes in the data, completely halting the flow until the missing packets can be recovered.

Over the years, there have been many enhancements to TCP's congestion control. From "selective acknowledgments" to "bottleneck bandwidth and round-trip propagation time," there have been many attempts to make TCP better at figuring out the right speed to send data to avoid drops and recover more quickly when drops occur. The result is more than a dozen variants of TCP in use today, none of which are efficient across the full scale of the internet.

For the most part, it has been left to applications to work around TCP's problems. For example, some have adopted multiplexed TCP sessions (many operations on a single TCP), while others use parallel TCP sessions (the opposite: a single operation split across many TCP sessions).

But ad-hoc workarounds have not been able to solve the fundamental efficiency problems that are costing billions of dollars per year.

➲ For the most part, it has been left to applications to work around TCP's problems;

ad-hoc workarounds have not been able to solve the fundamental efficiency problems that are costing billions of dollars per year.

A New Data Transport Model

The key limiting component of TCP in today's network is its virtual-circuit data model and the assumptions that data must always be delivered in order, that data must always be able to flow equally in both directions, and that congestion will rarely occur. But the internet is a best-effort packet-switched network, and as such it is inherently chaotic. Embracing that uncertainty rather than trying to abstract it away with a virtual circuit provides the flexibility and visibility for software to make the intelligent, real-time decisions required for full and efficient utilization of network hardware.

Ordered delivery of data is expensive, requiring buffering and stalls as TCP tries to juggle incoming packets. Because modern storage is fast and plentiful, and the most critical data transfers do not require ordered delivery, it is possible to deliver network data directly to files or memory as it arrives. This alleviates the hangs and speed oscillations that are so often associated with the internet. Smoother and more responsive flow control benefits the entire network, reducing congestion and recovering bandwidth from lost and delayed packets. Some data flows, like live video, require ordered delivery and buffering, so that capability must be available, just not mandatory.

A great deal of network overhead (the time and bandwidth consumed shepherding the actual data) can be eliminated by recognizing that most network communication is transactional: a small request pulling an arbitrary amount of data. For example, rather than the back-and-forth negotiation of TCP, simple yet secure transactions can begin with as few as one packet sent and one returned. Given a sufficiently lightweight implementation, more complex operations can be created by combining simple transactions as needed. Such modularity allows overhead to scale with the needs of each application, incurring only the minimum overhead required.

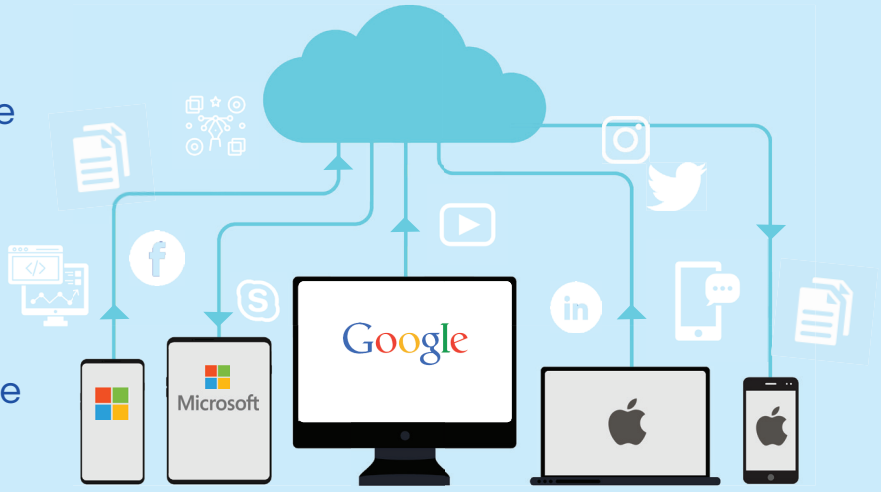
The problem of congestion control – figuring out how fast to send data on a best-effort network – is inherently difficult. All that a network node can really know about a path is when a packet was sent and when a reply was received (or not received). Everything else – speed, latency, packet loss – must be inferred from those simple events. Once freed from the constraints of a virtual circuit, the problem becomes much less difficult. Focusing solely on the arrival of data at a single point allows decisions about the network to be made without waiting for control data to cross that network. This independence means more timely and accurate adaptations to changing conditions.

Developing such novel transport algorithms is not easy, but it has been done. The looming challenge is deployment. Consider the state of IPv6, the "next generation" of the Internet Protocol. Despite nearly 20 years of production and a nominal launch date of 2012, as of 2019 fewer than 30 percent of internet users had adopted it. Network infrastructure change is slow. Fortunately, changing network standards is not necessary to create a new protocol.

In 1980, David Reed designed the User Datagram Protocol (UDP) so applications could create their own transport protocols without having to modify the TCP or IP layers. Since then, a number of open source and commercial efforts have built on top of UDP in search of greater efficiency. Nearly all of them have been constrained by the same virtual-circuit model as TCP, limiting their usefulness to well-funded enterprise networks and dedicated hobbyists. Even so, the freedom UDP gives to applications provides a path for deployment of completely new data models using existing infrastructure.

All of this still leaves the challenge that any new communication protocol requires software on both ends. Enterprise IT departments can implement mass adoption of such applications and technologies. So can a number of technology companies with global reach. Apple, Microsoft, and Alphabet (Google) each have vast ecosystems of devices. How much snappier would iPhones be if iCloud could be accessed with full network hardware utilization? What could Microsoft build if all its servers and desktops could communicate at high speed? How many more Google ads could be displayed if every Android device was querying Google's servers with a more efficient protocol? Google, in fact, has deployed its own UDP-based QUIC protocol for exactly this purpose, though its reported speed gains – about five percent – are modest at best.

How much snappier would iPhones be if iCloud could be accessed with full network hardware utilization? What could Microsoft build if all its servers and desktops could communicate at high speed?



Beyond the titans of computing, there are many other companies for whom improved data transport efficiency would mean a substantial boost to productivity, revenue, competitiveness, or a reduction in costs. Cloud vendors such as Amazon Web Services or Oracle are entirely dependent on the ability of their customers to efficiently move data in and out of their systems. While lacking the global reach of consumer device makers, the stakes are much higher for companies whose revenues are directly proportional to the data they process. Looking further, companies bringing new communication experiences to consumers are also in need of every competitive advantage. Facebook, Snapchat, Instagram, Twitter and dozens of upstarts will live or die by their ability to provide a better user experience.

Even if just one of these companies adopted a more efficient transport protocol, and even if it only made a 1% improvement in its revenue, the sheer scale would still represent billions of dollars in economic gain.

Decades of innovation have grown a worldwide economy that is completely dependent on real-time data transfer across a global packet-switched network. Yet almost all of that communication is funneled through a nearly 50-year-old virtual-circuit data model. The resulting inefficiencies are costing billions of dollars in underutilized resources and missed opportunities. Adoption of a more efficient transport protocol can be achieved on a per-application basis, without the need to develop new standards and consensus.

Such technology exists now, and a company with global reach could reap these rewards unilaterally or share them at their choosing.

➲ If just one of these companies adopted a more efficient transport protocol, and even if it only made a **1% improvement** in its revenue, the sheer scale would still represent **billions of dollars** in economic gain.

If you've read this far, you must be asking yourself: What does this mean for my company?

Or, what can be done now to impact my company's top line by 1% or more? It is a conversation worth having with the author, Dr. Seth Noble, one of the world's leading experts in accelerated data transport. He can answer these questions, as well as those you have not yet considered.

Seth can be reached directly at seth@dataexpedition.com.

About Seth Noble, PhD

Seth Noble, PhD, is the creator of the patented Multipurpose Transaction Protocol® (MTP™) technology and a top data transport expert. He is Founder and CEO of Data Expedition, Inc., with a dual BS-MS degree from Caltech, and a doctorate in computer science from the University of Oklahoma.

Forbes
Technology Council

As published in four parts on Forbes.com:

From Silicon To Memes: How Automated Calculators Grew Into A Global Communications Network, Part One

From Silicon To Memes: The Mechanisms Of Digital Communication, Part Two

From Silicon To Memes: The Cost Of Communicating With A Protocol From 1974, Part Three

From Silicon To Memes: A New Data Transport Model, Part Four