FEATURE AUGUST 2020

The Consensus Protocol

How a radical process, invented to build the ARPANET, spawned history's greatest tool for collaborative work

BY STEVE CROCKER

UCLA's Boelter Hall housed one of the four original ARPANET nodes.

Each March, July, and November, we are reminded that the Internet is not quite the mature, stable technology that it seems to be. We rely on the Internet as an essential tool for our economic, social, educational, and political lives. But when the Internet Engineering Task Force meets every four months at an open conference that bounces from continent to continent, more than 1,000 people from around the world gather with change on their minds. Their vision of the global network that all humanity shares is dynamic, evolving, and continuously improving. Their efforts combine with the contributions of myriad others to ensure that the Internet always works but is never done, never complete.

The rapid yet orderly evolution of the Internet is all the more remarkable considering the highly unusual way it happens: without a company, a government, or a board of directors in charge. Nothing about digital communications technology suggests that it should be self-organizing or, for that matter, fundamentally reliable. We enjoy an Internet that is both of those at once because multiple generations of network developers have embraced a principle and a

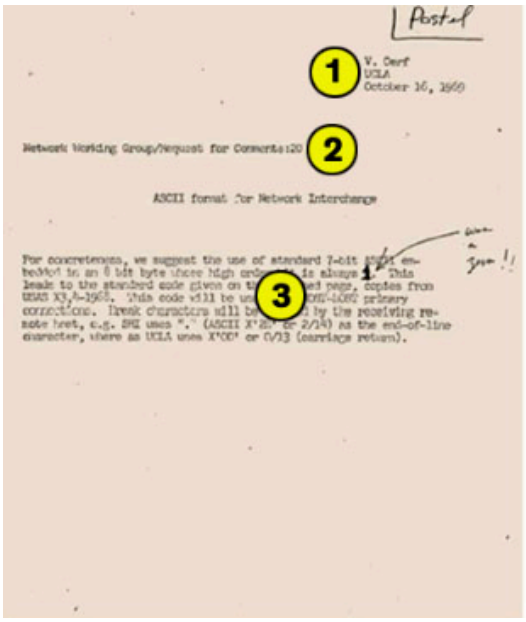
process that have been quite rare in the history of technology. The principle is that the protocols that govern how Internetconnected devices communicate should be open, expandable, and robust. And the process that invents and refines those protocols demands collaboration and a large degree of consensus among all who care to participate.

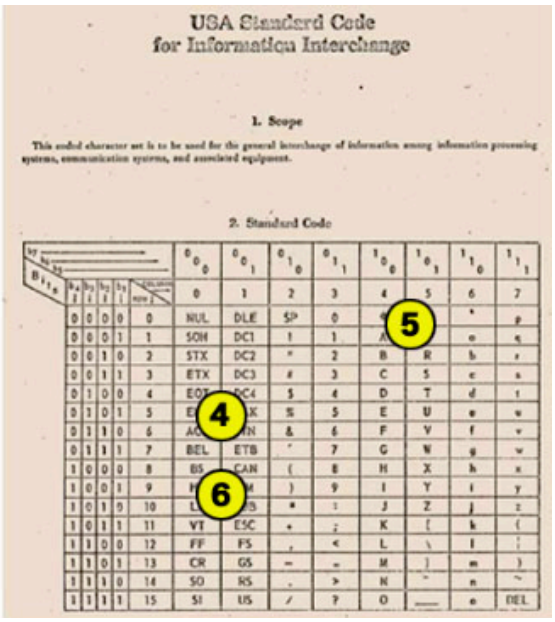
As someone who was part of the small team that very deliberately adopted a collaborative, consensus-based process to develop protocols for the ARPANET—predecessor to the Internet—I have been pleasantly surprised by how those ideas have persisted and succeeded, even as the physical network has evolved from 50-kilobit-per-second telephone lines in the mid-1960s to the fiber-optic, 5G, and satellite links we enjoy today. Though our team certainly never envisioned unforgeable "privacy passes" or unique identifiers for Internet-connected drones—two proposed protocols discussed at the task force meeting this past March—we did circulate our ideas for the ARPANET as technical memos among a far-flung group of computer scientists, collecting feedback and settling on solutions in much the same way as today, albeit at a much smaller scale.

We called each of those early memos a "Request for Comments" or RFC. Whatever networked device you use today, it almost certainly follows rules laid down in ARPANET RFCs written decades ago, probably including protocols for sending plain ASCII text (RFC 20, issued in 1969), audio or video data streams (RFC 768, 1980), and Post Office Protocol, or POP, email (RFC 918, 1984).

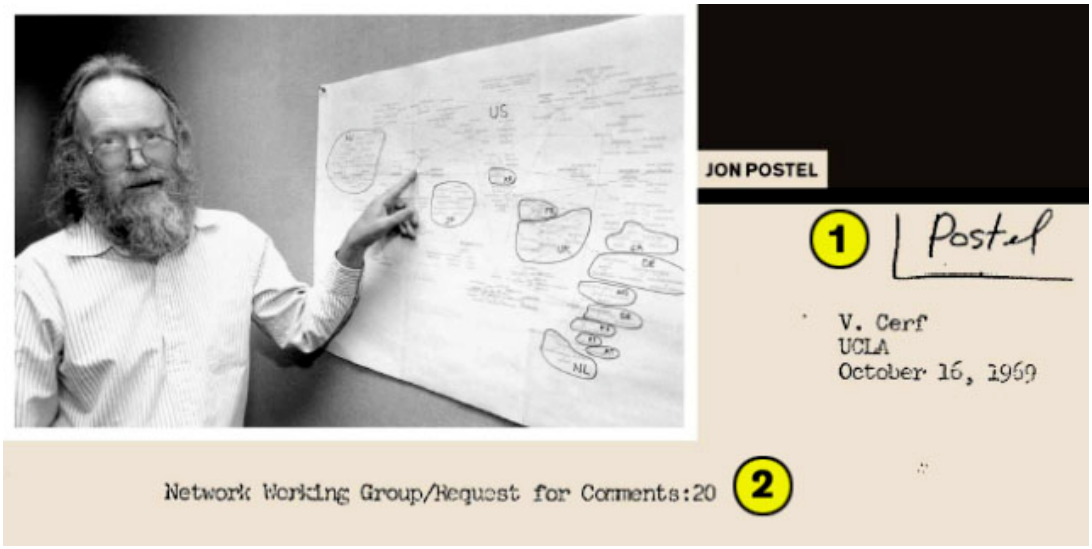
Of course, technology moves on. None of the computer or communication hardware used to build the ARPANET are crucial parts of the Internet today. But there is one technological system that has remained in constant use since 1969: the humble RFC, which we invented to manage change itself in those early days.

ANATOMY OF AN RFC



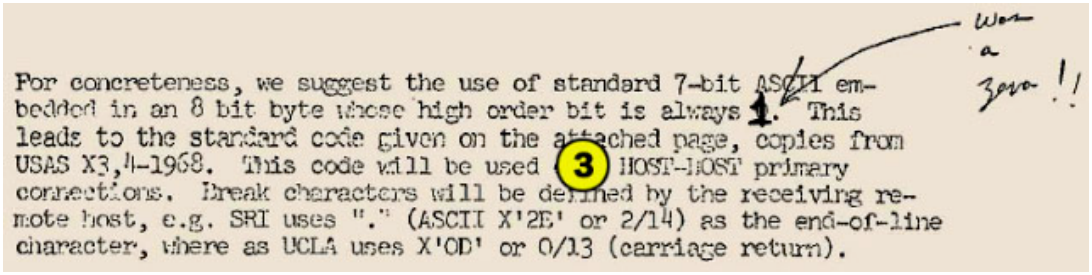


To manage the development of new protocols for the nascent ARPANET, the Network Working Group relied on Requests for Comments, or RFCs, which were agreed upon by consensus. RFC 20, issued in 1969, dealt with sending text over the ARPANET.

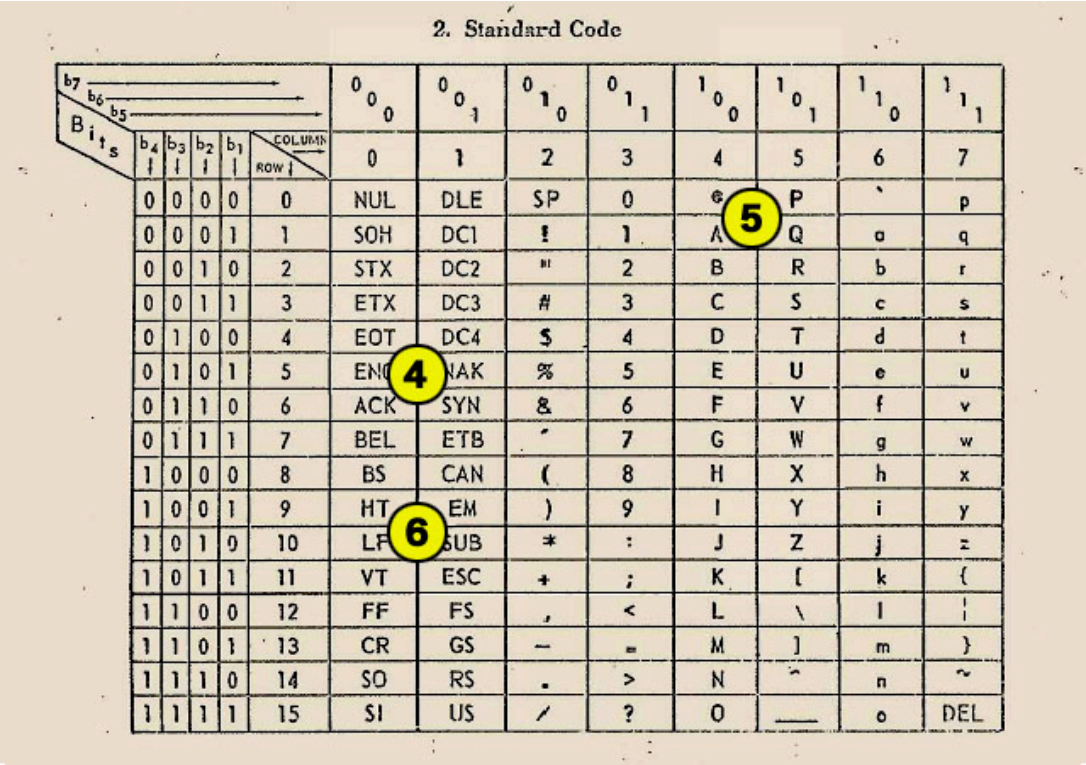


1. Jon Postel [above] was a high-school peer and UCLA colleague of Steve Crocker and Vint Cerf. He became known as the editor of the Request for Comments series, and he authored 197 RFCs. This is a scan of Postel's copy of RFC 20, in which Cerf proposed 7-bit ASCII as the standard text encoding for communication between ARPANET hosts. When Postel died in 1998, Cerf published his obit as RFC 2468.

2. Each RFC is assigned a serial number, usually after it's written to avoid gaps in the sequence. This one was the 20th to be circulated.



3. ARPANET connected a diverse set of machines, or hosts, whose operating systems used different numbers of bits to represent characters. The U.S. ASCII standard, which worked on both 7- and 8-bit systems, was thus ideal for host-to-host communication. Postel's copy of the RFC includes a suggestion to change the fixed high-order bit from 0 to 1. That change was not adopted.



2. Standard Code

Bits b_4	b_3	b_2	b_1	ROW ↓ / COLUMN → 0	0 0 0	0 0 1	0 1 0	0 1 1	1 0 0	1 0 1	1 1 0	1 1 1
					0	1	2	3	4	5	6	7
0	0	0	0	0	NUL	DLE	SP	0	@	P	`	p
0	0	0	1	1	SOH	DC1	!	1	A	Q	a	q
0	0	1	0	2	STX	DC2	"	2	B	R	b	r
0	0	1	1	3	ETX	DC3	#	3	C	S	c	s
0	1	0	0	4	EOT	DC4	$	4	D	T	d	t
0	1	0	1	5	ENQ	NAK	%	5	E	U	e	u
0	1	1	0	6	ACK	SYN	&	6	F	V	f	v
0	1	1	1	7	BEL	ETB	'	7	G	W	g	w
1	0	0	0	8	BS	CAN	(	8	H	X	h	x
1	0	0	1	9	HT	EM	)	9	I	Y	i	y
1	0	1	0	10	LF	SUB	*	:	J	Z	j	z
1	0	1	1	11	VT	ESC	+	;	K	[	k	{
1	1	0	0	12	FF	FS	,	<	L	\	l	|
1	1	0	1	13	CR	GS	-	=	M	]	m	}
1	1	1	0	14	SO	RS	.	>	N	^	n	~
1	1	1	1	15	SI	US	/	?	O	_	o	DEL

4. ASCII included 32 device and transmission control characters—such as ACK (acknowledge), EOT (end of transmission), and SYN (synchronous idle)—which were heavily used by teletype services, but not by ARPANET.

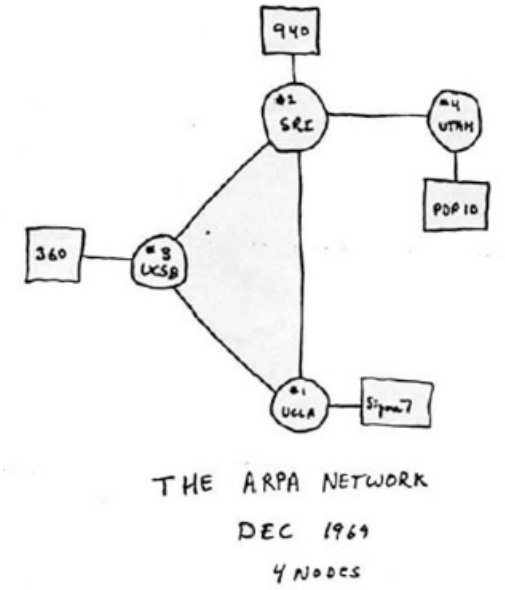
5. The RFC suggests that each of the 128 characters be represented by a 7-bit number, with three bits corresponding to its column in the table and four bits indicating the row. “A” is thus 1000001, also represented in column/row form as 4/1 or as 0x41 or decimal 65.

6. Some device control codes, such as HT (horizontal tabulation), were easily repurposed as technology evolved—in this case, to tabs. Other codes, such as CR (carriage return) and LF (line feed), were used inconsistently by different operating systems and took decades to sort out.

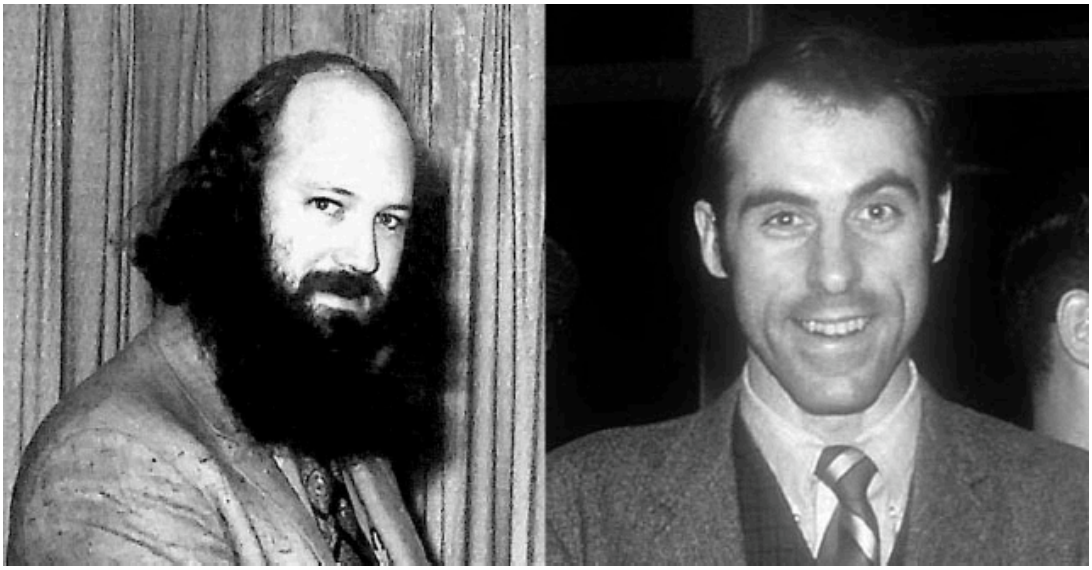
The ARPANET was far simpler than the Internet because it was a single network, not a network of networks. But in 1966, when the Pentagon’s Advanced Research Projects Agency (ARPA) started planning the idea of linking together completely different kinds of computers at a dozen or more research universities from California to Massachusetts, the project seemed quite ambitious.

It took two years to create the basic design, which was for an initial subnet that would exchange packets of data over dedicated telephone lines connecting computers at just four sites: the Santa Barbara and Los Angeles campuses of the University of California; Stanford Research Institute (SRI) in Menlo Park, Calif.; and the University of Utah in Salt Lake City. At each site, a router—we called them IMPs, for interface message processors—would chop outgoing blocks of bits into

smaller packets. The IMPs would also reassemble incoming packets from distant computers into blocks that the local "host" computer could process.



In December 1969, ARPANET had just four nodes.



Steve Crocker [left] and Vint Cerf [right] were grad students when they began working on the ARPANET project at UCLA in 1968.

In the tumultuous summer of 1968, I was a graduate student spending a few months in the computer science department at UCLA, where a close friend of mine from high school, Vint Cerf, was studying. Like many others in the field, I was much more interested in artificial intelligence and computer graphics than in networking. Indeed, some principal investigators outside of the first four sites initially viewed the ARPANET project as an intrusion rather than an opportunity. When ARPA invited each of the four pilot sites to send two people to a kickoff meeting in Santa Barbara at the end of August, Vint and I drove up from UCLA and discovered that all of the other attendees were also graduate students or staff members. No professors had come.

Almost none of us had met, let alone worked with, anyone from the other sites before. But all of us had worked on timesharing systems, which doled out chunks of processing time on centralized mainframe computers to a series of remotely connected users, and so we all had a sense that interesting things could be done by connecting distant computers and getting their applications to interact with one another. In fact, we expected that general-purpose interconnection of computers would be so useful that it would eventually spread to include essentially every computer. But we certainly did not anticipate how that meeting would launch a collaborative process that would grow this little network into a critical piece of global infrastructure. And we had no inkling how dramatically our collaboration over the next few years would change our lives.

After getting to know each other in Santa Barbara, we organized follow-up meetings at each of the other sites so that we would all have a common view of what this eclectic network would look like. The SDS Sigma 7 computer at UCLA would be connecting to a DEC PDP-10 in Utah, an IBM System/360 in Santa Barbara, and an SDS 940 at SRI.

We would be a distributed team, writing software that would have to work on a diverse collection of machines and operating systems—some of which didn't even use the same number of bits to represent characters. Co-opting the name of the ARPA-appointed committee of professors that had assigned us to this project, we called ourselves the Network Working Group.

A team at Bolt, Beranek and Newman built the interface message processors, or IMPs, for each ARPANET site.

We had only a few months during the autumn of 1968 and the winter of 1969 to complete our theoretical work on the general architecture of the protocols, while we waited for the IMPs to be built in Cambridge, Mass., by the R&D company Bolt, Beranek and Newman (BBN).

Our group was given no concrete requirements for what the network should do. No project manager asked us for regular status reports or set firm milestones. Other than a general assumption that users at each site should be able to remotely log on and transfer files to and from hosts at the other sites, it was up to us to create useful services.

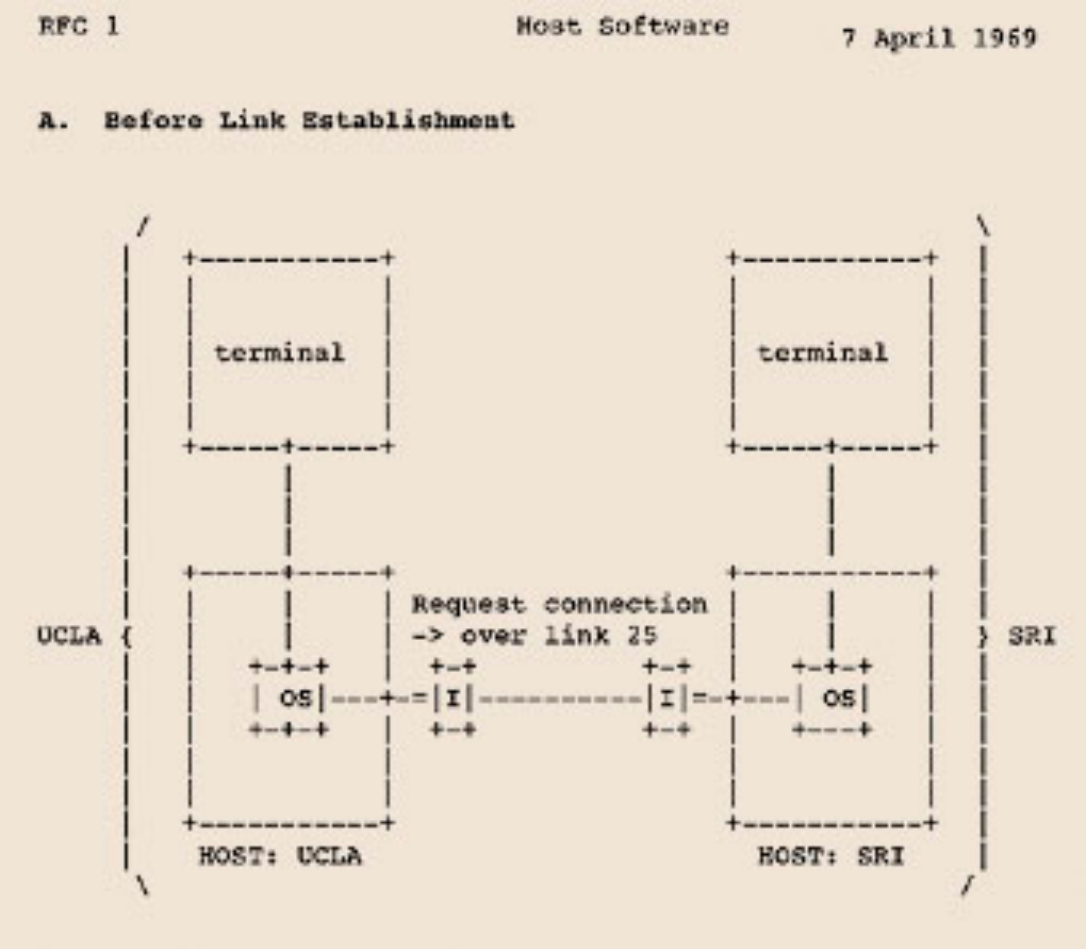
Through our regular meetings, a broader vision emerged, shaped by three ideas. First, we saw the potential for lots of interesting network services. We imagined that different application programs could exchange messages across the network,

for example, and even control one another remotely by executing each other's subroutines. We wanted to explore that potential.

Second, we felt that the network services should be expandable. Time-sharing systems had demonstrated how you could offer a new service merely by writing a program and letting others use it. We felt that the network should have a similar capacity.

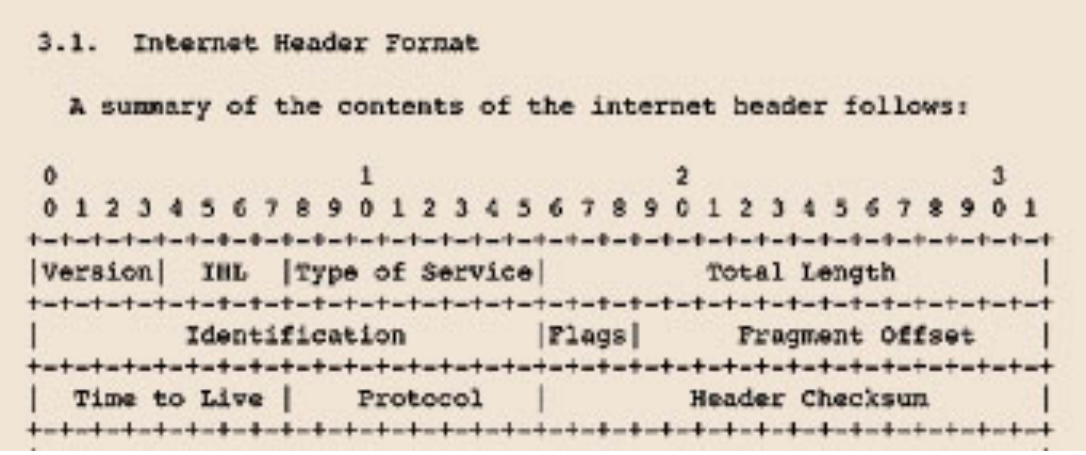
3 RFCs to Know

1 Host Software



Issued in April 1969 and written by Steve Crocker, then a grad student at UCLA, RFC 1 and its companion, RFC 2, by Bill Duvall of SRI, describe various aspects of the software used to connect the ARPANET's host computers and IMPs. The fluidity of the RFC process is apparent at the start. "Very little of what is here is firm, and reactions are expected," Crocker wrote.

760 Internet Protocol



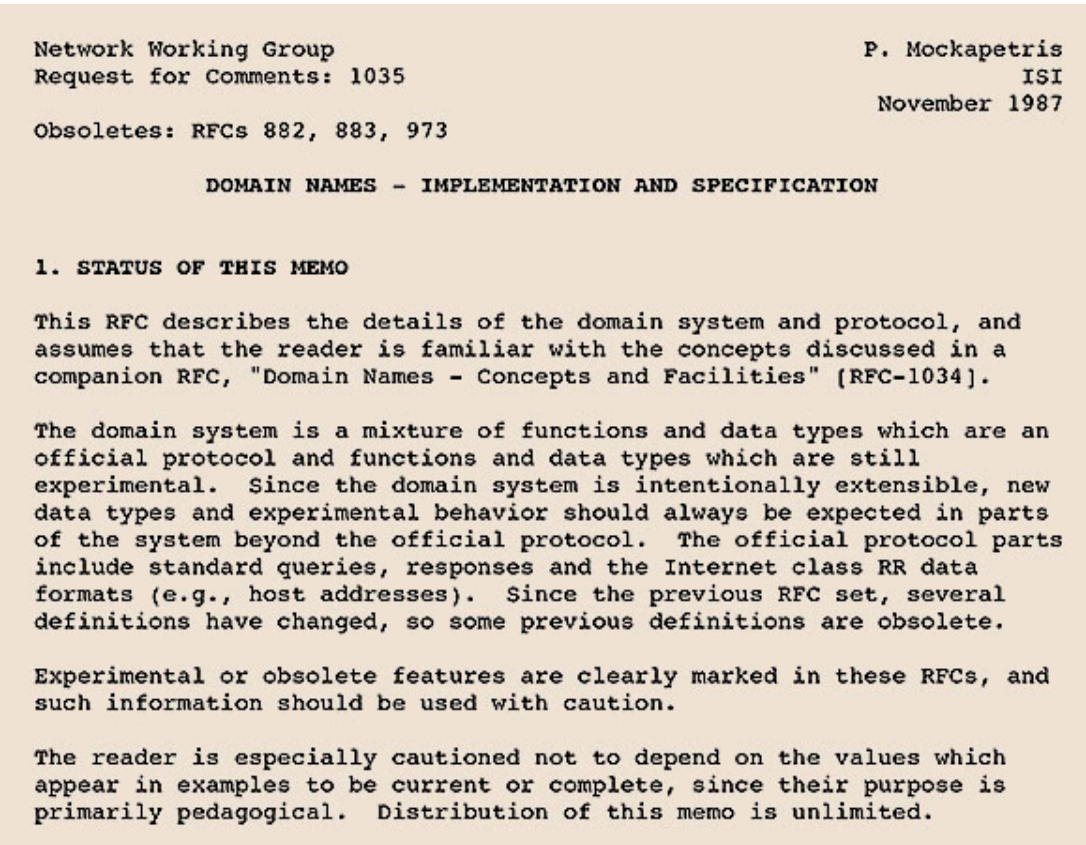
3.1. Internet Header Format

A summary of the contents of the internet header follows:

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|Version|  IHL  |Type of Service|          Total Length         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|         Identification        |Flags|      Fragment Offset    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|  Time to Live |    Protocol   |         Header Checksum       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Issued in January 1980, this RFC outlines one of the foundational elements of Internet architecture: Internet Protocol. To this day, most Internet data is routed from one network to another in IPv4 packets, as laid out in this RFC. Because IPv4 addresses are limited to 4 billion hosts, a new IPv6 protocol was introduced as an Internet standard in 1995.

1035 Domain Names



Network Working Group — P. Mockapetris
Request for Comments: 1035 — ISI
November 1987

Obsoletes: RFCs 882, 883, 973

DOMAIN NAMES - IMPLEMENTATION AND SPECIFICATION

1. STATUS OF THIS MEMO

This RFC describes the details of the domain system and protocol, and assumes that the reader is familiar with the concepts discussed in a companion RFC, "Domain Names - Concepts and Facilities" (RFC-1034).

The domain system is a mixture of functions and data types which are an official protocol and functions and data types which are still experimental. Since the domain system is intentionally extensible, new data types and experimental behavior should always be expected in parts of the system beyond the official protocol. The official protocol parts include standard queries, responses and the Internet class RR data formats (e.g., host addresses). Since the previous RFC set, several definitions have changed, so some previous definitions are obsolete.

Experimental or obsolete features are clearly marked in these RFCs, and such information should be used with caution.

The reader is especially cautioned not to depend on the values which appear in examples to be current or complete, since their purpose is primarily pedagogical. Distribution of this memo is unlimited.

Written by Paul Mockapetris and issued in November 1987, this RFC and RFC 1034 describe how to map human-readable host names like spectrum.ieee.org to network addresses like 99.86.33.6. Mockapetris's and Jon Postel's domain name system (DNS) distributes lookup tables across myriad servers, allowing the network to scale to global size.

Finally, we recognized that the network would be most useful if it were agnostic about the hardware of its hosts. Whatever software we wrote ought to support any machine seamlessly, regardless of its word length, character set, instruction set, or architecture.

We couldn't translate these ideas into software immediately because BBN had yet to release its specification for the interface to the IMP. But we wanted to get our thoughts down on paper. When the Network Working Group gathered in Utah in March 1969, we dealt out writing assignments to one another. Until we got the network running and created an email protocol, we would have to share our memos through the U. S. Mail. To make the process as easy and efficient as possible, I kept a numbered list of documents in circulation, and authors mailed copies of memos they wrote to everyone else.

Tentatively, but with building excitement, our group of grad students felt our way through the dark together. We didn't even appoint a leader. Surely at some point the "real experts"—probably from some big-name institution in the Northeast or Washington, D.C.—would take charge. We didn't want to step on anyone's toes. So we certainly weren't going to call our technical memos "standards" or "orders." Even "proposals" seemed too strong—we just saw them as ideas that we were communicating without prior approval or coordination. Finally, we settled on a term suggested by Bill Duvall, a young programmer at SRI, that emphasized that these documents were part of an ongoing and often preliminary discussion: Request for Comments.

The first batch of RFCs arrived in April 1969. What was arguably one of our best initial ideas was not spelled out in these RFCs but only implicit in them: the agreement to structure protocols in layers, so that one protocol could build on another if desired and so that programmers could write software that tapped into whatever level of the protocol stack worked best for their needs.

We started with the bottom layer, the foundation. I wrote RFC 1, and Duvall wrote RFC 2. Together, these first two memos described basic streaming connections between hosts. We kept this layer simple—easy to define and easy to implement. Interactive terminal connections (like Telnet), file transfer mechanisms (like FTP), and other applications yet to be defined (like email) could then be built on top of it.

That was the plan, anyway. It turned out to be more challenging than expected. We wrestled with, among other things, how to establish connections, how to assign addresses that allowed for multiple connections, how to handle flow control, what to use as the common unit of transmission, and how to enable users to interrupt the remote system. Only after multiple iterations and many, many months of back-and-forth did we finally reach consensus on the details.

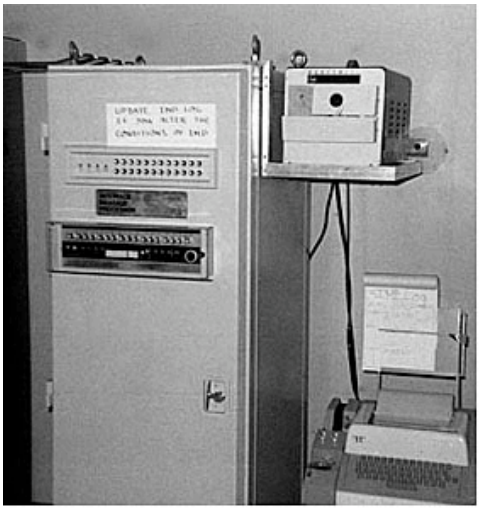
Some of the RFCs in that first batch were more administrative, laying out the minimalist conventions we wanted these memos to take, presenting software

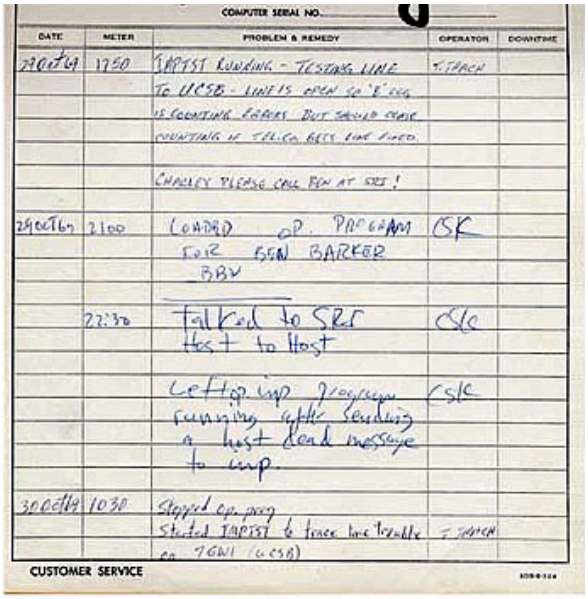
testing schedules, and tracking the growing mailing list.

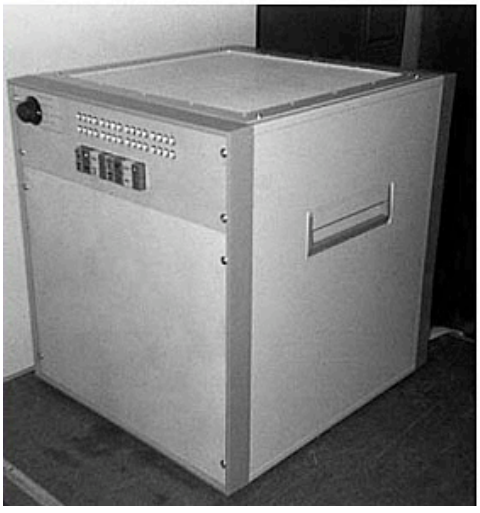
Others laid out grand visions that nevertheless failed to gain traction. To my mind, RFC 5 was the most ambitious and interesting of the lot. In it, Jeff Rulifson, then at SRI, introduced a very powerful idea: downloading a small application at the beginning of an interactive session that could mediate the session and speed things up by handling "small" actions locally.

As one very simple example, the downloaded program could let you edit or auto-complete a command on the console before sending it to the remote host. The application would be written in a machine-agnostic language called Decode-Encode Language (DEL). For this to work, every host would have to be able to run a DEL interpreter. But we felt that the language could be kept simple enough for this to be feasible and that it might significantly improve responsiveness for users.

Aside from small bursts of experimentation with DEL, however, the idea didn't catch on until many years later, when Microsoft released ActiveX and Sun Microsystems produced Java. Today, the technique is at the heart of every online app.







October 1969, Bill Duvall at the Stanford Research Institute received the ARPANET's first message, sent by Charley Kline via UCLA's interface message processor and a second interface between the IMP and the host that

message processor and a second interface between the IMP and the host that

had been designed by grad student Mike Wingfield. The task was dutifully logged at 22:30: “Talked to SRI Host to Host.”

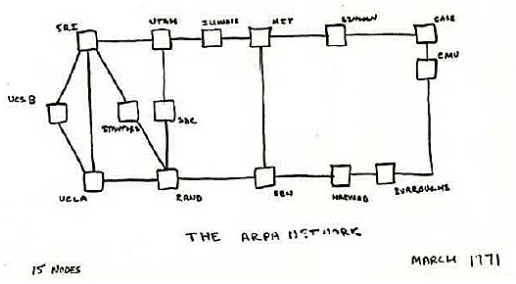
The handful of RFCs we circulated in early 1969 captured our ideas for network protocols, but our work really began in earnest that September and October, when the first IMPs arrived at UCLA and then SRI. Two were enough to start experimenting. Duvall at SRI and Charley Kline at UCLA (who worked in Leonard Kleinrock’s group) dashed off some software to allow a user on the UCLA machine to log on to the machine at SRI. On the evening of 29 October 1969, Charley tried unsuccessfully to do so. After a quick fix to a small glitch in the SRI software, a successful connection was made that evening. The software was adequate for connecting UCLA to SRI, but it wasn’t general enough for all of the machines that would eventually be connected to the ARPANET. More work was needed.

By February 1970, we had a basic host-to-host communication protocol working well enough to present it at that spring’s Joint Computer Conference in Atlantic City. Within a few more months, the protocol was solid enough that we could shift our attention up the stack to two application-layer protocols, Telnet and FTP.

Rather than writing monolithic programs to run on each computer, as some of our bosses had originally envisioned, we stuck to our principle that protocols should build on one another so that the system would remain open and extensible. Designing Telnet and FTP to communicate through the host-to-host protocol guaranteed that they could be updated independently of the base system.

By October 1971, we were ready to put the ARPANET through its paces. Gathering at MIT for a complete shakedown test—we called it “the bake-off”—we checked that each host could log on to every other host. It was a proud moment, as well as a milestone that the Network Working Group had set for itself.

And yet we knew there was still so much to do. The network had grown to connect 23 hosts at 15 sites. A year later, at a big communications conference in Washington, D.C., the ARPANET was demonstrated publicly in a hotel ballroom. Visitors were able to sit down at any of several terminals and log on to computers all over the United States.



By March 1971, the ARPANET had grown to 15 nodes.

Year after year, our group continued to produce RFCs with observations, suggested changes, and possible extensions to the ARPANET and its protocols. Email was among those early additions. It started as a specialized case of file transfer but was later reworked into a separate protocol (Simple Mail Transfer Protocol, or SMTP, RFC 788, issued in 1981). Somewhat to the bemusement of both us and our bosses, email became the dominant use of the

ARPANET, the first “killer app.”

Email also affected our own work, of course, as it allowed our group to circulate RFCs faster and to a much wider group of collaborators. A virtuous cycle had begun: Each new feature enabled programmers to create other new features more easily.

Protocol development flourished. The TCP and IP protocols replaced and greatly enhanced the host-to-host protocol and laid the foundation for the Internet. The RFC process led to the adoption of the Domain Name System (DNS, RFC 1035, issued in 1987), the Simple Network Management Protocol (SNMP, RFC 1157, 1990), and the Hypertext Transfer Protocol (HTTP, RFC 1945, 1996).

In time, the development process evolved along with the technology and the growing importance of the Internet in international communication and commerce. In 1979, Vint Cerf, by then a program manager at DARPA, created the Internet Configuration Control Board, which eventually spawned the Internet Engineering Task Force. That task force continues the work that was originally done by the Network Working Group. Its members still discuss problems facing the network, modifications that might be necessary to existing protocols, and new protocols that may be of value. And they still publish protocol specifications as documents with the label “Request for Comments.”

And the core idea of continual improvement by consensus among a coalition of the willing still lives strong in Internet culture. Ideas for new protocols and changes to protocols are now circulated via email lists devoted to specific protocol topics, known as working groups. There are now about a hundred of these groups. And when they meet at the triannual conferences, the organizers still don’t take votes: They ask participants to hum if they agree with an idea, then take the sense of the room. Formal decisions follow a subsequent exchange over email.

Drafts of protocol specifications are circulated as “Internet-Drafts,” which are intended for discussion leading to an RFC. One discussion that was recently begun on new network software to enable quantum Internet communication, for example, is recorded in an RFC-like Internet-Draft.

And in an ironic twist, the specification for this or any other new protocol will appear in a Request for Comments only *after* it has been approved for formal adoption and published. At that point, comments are no longer actually requested.

POST YOUR COMMENTS AT spectrum.ieee.org/arpanetrfc-aug2020