KERMIT NEWS

ISSN 0899-9309

Number 4

June 1990

Contents

***Kermit News**, ISSN 0899-9309, is published periodically free of charge by Kermit Development and Distribution, Columbia University Center for Computing Activities, 612 West 115th Street, New York, NY 10025, USA. Contributed articles are welcome. Material in **Kermit News** may be quoted or reproduced without permission, but with proper attribution. And be sure to send us a copy!*

Editor: *Christine M. Gianone*
E-Mail: `cmg@watsun.cc.columbia.edu` *or* `kermit@cuvma.bitnet`

The Kermit file transfer protocol is named after Kermit the Frog, star of the television series The Muppet Show, used by permission of Henson Associates, Inc.

Editor's Notes

Christine M. Gianone, Manager
Kermit Development and Distribution
Columbia University, New York City, USA

Publication of *Kermit News* began in 1986 to keep Kermit users up to date on developments in the Kermit protocol, software, and documentation. Response has been overwhelming, especially from those without access to Columbia University's electronic newsletter, the Info-Kermit Digest.

This is the fourth issue of *Kermit News*. This and future issues will no longer have a Volume number. Welcome to *Kermit News* Number 4! What this newsletter lacks in frequency it should make up for in news.

A lot has happened since the last issue was printed in June 1988. This issue features important new releases of Kermit software for most major computers and operating systems, including MS-DOS Kermit 3.0, C-Kermit 5A, and IBM Mainframe Kermit-370 4.2, a ''new image'' in documentation to help promote Kermit as a serious communications software product, a major extension to the Kermit protocol to support international character sets, implementation of sliding windows and local area network support in several major Kermit programs, newsworthy events including a visit to the USSR.

What Are You Doing? If you have an interesting Kermit story to tell, please send it in. Even if you don't think it's interesting, we'd still like to know how you are using Kermit. Is it saving you money, time, aggravation? Is it making your life better in some small way? Let us know!

Where Do Kermit Programs Come From? Kermit Development and Distribution is managed from an obscure little office on the outskirts of the Columbia University campus in northern Manhattan. Volunteer programmers from all over the world, usually working on their own time, contribute new or updated Kermit programs with enthusiasm, generosity, and dedication you won't see anywhere else in the software industry.

Support Your Local Kermit! Grants and donations are always welcome to help advance the Kermit protocol and develop and distribute Kermit software. Our office must remain self-supporting to continue operating.

Special thanks to all the companies that have contributed $2000 or more (tax deductible) in equipment or cash in the past two years:

- Apple Computer Corporation of Cupertino, California, USA, for an equipment grant to further Macintosh Kermit development.
- OMNI Solutions Inc. of Los Altos, California, USA, for a generous cash donation.

And thanks to those of you who noticed the unobtrusive little ''Contribution'' box at the end of our order form and donated smaller amounts. Page 30 lists all of you.

It's Not Just Academic!

There are three widespread misconceptions about Kermit. The first is that it is ''just'' a file transfer protocol that one finds implemented in commercial software packages like Crosstalk, Smartcom, etc, but it is not a software package itself. Not true: Kermit *programs* for hundreds of different computers and operating systems are now available from Columbia University. Many of these programs are full-function communication software packages comparable to the high-priced commercial alternatives.

The second misconception is that Kermit software is *academic*, used only in universities. True, the Kermit protocol was born at a university but it has since travelled far and wide −to every part of the world, to every sector of the economy. This brings us to the third myth: that Kermit is a toy −an insignificant, little-used product. In this issue of *Kermit News*, we will try to lay these three misconceptions to rest.

Corporate Use of Kermit

Since Columbia first began to distribute Kermit material in 1981, tens of thousands of tapes, diskettes, manuals, and books have been shipped to every state in the USA and more than 70 other countries in all parts of the world. Of these, about 23% have gone to universities and schools, 24% to government or public agencies, and 46% to corporations. The remainder have gone to individuals or customers in other categories. Thus, the private sector leads all others in Kermit orders.

Comparison of Columbia's Kermit customer database with the April 1989 Fortune 500 listing shows:

Fortune	15	100%
Fortune	50	94%
Fortune	100	80%
Fortune	200	65%
Fortune	300	56%
Fortune	400	49%
Fortune	500	42%

These figures would be impressive for a commercial venture, but they are extraordinary for a small operation consisting of a handful of people, without a marketing or advertising budget. And they do not reflect the many companies that have obtained Kermit from other sources. A single copy of Kermit may be reproduced many times over, at vast savings to a large organization compared to commercial, licensed software.

The Fortune 500 figures are not totally accurate because many of these companies are vague corporate entities whose names don't show up in the Kermit Distribution database because the orders come from the actual companies that they hold.

Another measure of Kermit's penetration into the corporate sector is the Datamation 100 Leading USA IS Companies (listed in Datamation, June 15, 1989):

Datamation	20	100%
Datamation	50	96%
Datamation	100	80%

and the Datamation 50 Leading Non-USA IS Companies:

Datamation	20	85%
Datamation	50	58%

The Kermit protocol has been incorporated into several *hundred* different commercial products, ranging from the well-known communication software programs, to laboratory and manufacturing equipment (see p.23), to data communication equipment (like the Telebit Trailblazer Modem), to spread-spectrum radio applications (p.24), to scientific calculators (like the new HP-48SX), to fast-food-chain cash registers (page 22).

Government Use of Kermit

Local, state, province, and national government agencies in the US, Canada, and other countries are among the biggest customers. Kermit software has been shipped to virtually every US and Canadian national government agency and to most major government contractors, as well as to thousands of county governments, school districts, state agencies, etc, and to international organizations like the European Parliament, the European Space Agency, the International Atomic Energy Agency, the North Atlantic Treaty Organization, the Organization of American States, the United Nations, and the World Bank.

It has been reported repeatedly over the years that Kermit is a requirement in major US government Requests for Proposals (RFPs). For example, it was reported from the headquarters of the US Air Force Logistic Command at McClellan Air Force Base in California that Kermit file transfer protocol is a requirement for all Air Force terminal emulation software, and that the Kermit software itself is in use on thousands of computers throughout the Air Force.

A typical military Request for Proposals is US Army RFP for Small Multi-User Computer, DAEA26-87-R-0007, SMC, which invites vendors to submit bids for about 20,000 systems. Appendix VIII, Section 3.4.14b, lists support for Kermit protocol to upload and download ASCII and binary files as one of the requirements. Suppose one of these computers costs \$5000. That's a one hundred million dollar contract. Now suppose (we don't know this) that the Army decides to equip these computers with Kermit software rather than a commercial package costing (say) \$250. If this happens, Kermit has just saved the US taxpayers \$5,000,000.

A similar story can be told of the US Army Recruiter Workstation project, currently under development by the Ohio National Guard, in which PCs at recruitment offices worldwide will communicate enlistment applicant data to central host computers using Kermit software. How many recruiting offices are there? Multiply that number by the software cost savings, and let's hope the result is a sizeable ''peace dividend''.

With the increasing popularity of PC networks, our governments have an even greater money-saving potential in Kermit software. The popular commercial PC communication software packages can be used in the PC network environment only if you purchase special ''network editions'' at high cost, typically \$400-700 per unit. MS-DOS Kermit 3.0 operates in the same environments at negligible cost (see the article on p.4). Hey, taxpayers, it's our money!

MS-DOS Kermit 3.0 Debuts

Version 3.0 of MS-DOS Kermit for the IBM PC, PS/2, and compatibles was released in January 1990 by Professor Joe R. Doupnik of the Center for Atmospheric and Space Sciences and Department of Electrical Engineering of Utah State University in Logan, Utah, USA, in cooperation with Columbia University, and with contributions from many others in the international academic and corporate sectors. The major new features of version 3.0 are listed below; separate articles go into in more detail.

- Transfer of text files in international character sets using a new extension to the Kermit protocol (see page 16).
- Emulation of the DEC VT320 terminal, including soft function keys and support for a wide variety of international character sets during terminal connection.
- Sliding window packet protocol for improved file transfer performance over public data networks and long distance satellite connections, fully integrated with long packets (see p.12).
- Expanded support for local area networks, including IBM Netbios, StarLAN, Novell NASI/NACS and TES, IBM EBIOS/LANACS, DECnet LAT and CTERM, Ungermann-Bass Net/One, Intel OpenNET, 3COM BAPI, and TCP/IP with vendor-provided utilities (see page 4).
- Enhanced Tektronix graphics terminal emulation with DEC VT340 sixel graphics extensions, suitable for use with mainframe WordPerfect versions 5 and 4.2 and other applications. Support for color, line patterns, rectangle fill patterns. Graphics screens may be saved in TIFF 5.0 format for importation into applications like WordPerfect, Pagemaker, and Ventura Publisher, and they may be printed on a variety of printers.

Other additions include instantaneous screen rollback and increased capacity for saved screens, saving of cleared screens, improved Microsoft Windows compatibility, a visual bell for deaf users, selectable screen writing direction (for languages like Hebrew and Arabic), support for the high-performance features of the PS/2's 16550A communications chip, RTS/CTS handshaking for use with half-duplex modems and radio transmitters (see p.24), and support for 75/1200 bps split-speed operation

Version 3.0 continues to support advanced Kermit protocol features such as long packets, file attributes, and a secure server mode, as well as command files and macros, a (now even more powerful) script programming language, VT52/100/102 emulation, key redefinition, screen rollback and capture, printer control, compatibility with Microsoft Windows, and more.

The program, on a 5.25-inch PC diskette, is packaged together with the new book *Using MS-DOS Kermit* by Christine M. Gianone, Digital Press (1990). This package may be ordered by mail from Columbia (see the order form in back), purchased in computer bookstores, or ordered by phone from Digital Press for $29.95 by calling 1-800-343-8321 (order number EY-C204E-DP, USA only) and providing a credit card number.

Since the last issue of *Kermit News* there were also other MS-DOS Kermit releases. Pre-3.0 releases included 2.31, 2.32, or 2.32/A for the IBM family, the DEC Rainbow, the Wang PC, the Victor 9000, Grid Compass, HP-110, HP-150, HP Portable PC, NEC PC 9801, etc. The non-IBM-compatible versions necessarily lag one or more releases behind the IBM version. These are all on Tape A, as well as test versions of 3.0 for the Rainbow, Grid, and HP PCs.

MS-DOS Kermit 3.0 Early Reviews

''I *LOVE* Kermit! It is wonderful to have a free communication package that I can distribute to users just getting started dailing in to our VAX 8800 and/or online library catalog.'' − *Prof. Joe St Sauver, University of Oregon*

''This version is AMAZING!!!! . . . Until today, I didn't realize how powerful the DECnet interface is . . . We have modem servers around the network and never thought that a PC could Kermit over one of those modems −but SET PORT DECNET does the trick. Nice job!!!!''− *Chris Lent, Robert Weiner, Cooper Union*

''Initial impressions of MS-DOS Kermit 3.0 are very good indeed. Praise and congratulations to all those working on this project. Running vs C-Kermit on our VAXen, and using 1000-byte packets, it typically delivers twice the net data rate provided by Linkware.'' − *Roger Wallace, Raytheon Co., Research Div.*

''I have been testing and sending a text file in Swedish IBM ASCII on the PC to UNIX with Swedish 7-bit ASCII [using MS-DOS Kermit 3.0 and C-Kermit 5A]. It works and that is very GOOD!'' − *Bo Kullmar, Central Bank of Sweden*

''MS-DOS Kermit is the only PC communication program that works 100 percent with speech devices and software used by the blind.'' − *Rick Hayner, Portland, Michigan*

MS-DOS Kermit 3.0 on Local Area Networks

Joe R. Doupnik and Christine M. Gianone

Many of us are accustomed to using Kermit on our PCs to access timesharing computers or to dial up data services. But MS-DOS Kermit 3.0 supports another mode of communication that is becoming increasingly popular: the local area network, or LAN.

IBM PCs and PS/2s can be connnected to LANs based on Ethernet, Token Ring, or other media using a variety of software and protocols: AT&T StarLAN/StarGROUP, DECnet (including LAT and CTERM), IBM EBIOS/LANACS, Intel OpenNET, Netbios, Novell NASI/NACS and TES; TCP/IP Telnet from FTP Inc., Novell, and Interlan; 3COM BAPI, and Ungermann-Bass Net/One. MS-DOS Kermit 3.0 contains some form of support for all of these, and more.

Novell Networks

To get an idea of what Kermit can do on a LAN, let's look at its support for Novell and Netbios-based networks. MS-DOS Kermit 3.0 can operate with Novell products in at least five ways: with a file server, with an asynchronous communication server, station-to-station, with NetWare/VMS, and through a TCP/IP gateway.

File and Print Servers

The use of Kermit with a file server is especially easy because the file server looks like a local PC device to Kermit: as an additional set of disk drive letters and/or as extra printers. Special procedures are built into Kermit to assist printing to a network printer while Kermit is attached to a host via a high speed communications port. Network printing is achieved by using the Novell utility CAPTURE to redirect a PC printer channel to the file server.

Asynchronous Communication Servers

An asynchronous communication server (ACS) is a device on the PC network that houses serial communication ports and/or modems that all the PCs on the network can share, using them as if they were locally attached. On a Novell network, this is done by running the Novell NASI utility on your PC. The asynchronous server runs the matching part, named NACS. Communication programs like Kermit that understand the NASI protocol can use it to communicate with the ACS.

Until the release of MS-DOS Kermit 3.0, the use of asynchrounous communication servers was an expensive proposition because of the many copies required of ''network versions'' of costly commercial packages. Now Kermit can fill this need at little or no cost. Just give it the commands SET PORT NOVELL and CONNECT, and you're communicating with the ACS. You can even maintain multiple simultaneous connections and switch among them with a single keystroke.

Station-To-Station

Station-to-station communication is also possible with Kermit, using the SET PORT NETBIOS command. Any two PCs on the Novell (or other Netbios-based) network can transfer, print, and manage files between themselves using Kermit's normal file transfer commands. This is a useful option when the file server is unavailable, does not provide the flexibility that you need, or when its disk is full.

Terminal Emulation

VAX/VMS computers can join Novell networks too, and PCs with appropriate software can log in to them directly over the Novell network. Novell's TES program and shell execute the low-level protocols, and MS-DOS Kermit rides on top of them if you give it the commands SET PORT BIOS1 and CONNECT. Multiple sessions are available via TES's hot-key menu to place connections on hold, or hangup old ones, or make new ones. To use this feature, you must assign the new Kermit verb `\Knethold` to a key (using Kermit's SET KEY command), and then press this key whenever you want to switch sessions. The same `\Knethold` key is used by several packages to awaken the external control program and, for example, change modem speed or change sessions. Similarly, the normal commands to send a BREAK and to HANGUP a connection are converted to do the operation over the network if the network supports the concepts.

TCP/IP Access

Finally, if your Novell network has a TCP/IP gateway such as the one sold by Interlan, and if the maker of the gateway has provided appropriate driver software, MS-DOS Kermit 3.0 can connect at very high speeds to any host on the TCP/IP network. For example, to get through the Interlan gateway using Kermit, run Kermit from Interlan's TELNET program by issuing the command:

```
telnet hostname Int14h-kermit
```

and then give Kermit the commands SET PORT BIOS1 and CONNECT. Then you will have a very high-speed Telnet connection to the Internet host you have selected, but with Kermit's user interface (screen rollback, key mapping, character sets, and so on).

Other Networks

If you have a PC network other than Novell, chances are that MS-DOS Kermit 3.0 will work in your environment too.

IBM EBIOS/LANACS is another Asynchronous Communications Server protocol. To use it with Kermit, load `EBIOS.SYS` in your `CONFIG.SYS` file, run the REDIRECT program, and tell Kermit to SET PORT EBIOS 1 (or 2, 3, or 4). Then CONNECT, and you're on your way. A BREAK may be sent and the HANGUP command also works to the server's RS-232 device.

Ungermann-Bass has a local area network method named Net/One. To use Kermit activate the network and then Kermit. Command SET PORT UB followed by CONNECT invokes the Net/One command interface to form connections to hosts. Kermit keyboard verbs `\Knethold` and `\KBreak` are useful to regain attention of the command interface again, or to send a BREAK signal across the network, respectively.

Intel OpenNET and MS-DOS Kermit work together by running the OpenNET NetBios program and then Kermit. The Kermit Command SET PORT NETBIOS *hostname* starts a connection to the remote host.

3Com BAPI is the name for the 3Com Bridge Applications Programmer Interface to 3Com networks. It simulates a serial port with the fast network path. Start the network and then Kermit. Assign `\Knethold` to a convenient key with the Kermit SET KEY command. The commands SET PORT 3COM and then CONNECT start the process and the BAPI command menu should appear to select a host.

StarLAN and StarGROUP are AT&T's networking products, and MS-DOS Kermit has implicit support for this network method too. First run the NetBios program for the network and then Kermit. The command SET PORT NETBIOS *nodename* will then start a connection to login to a remote Unix host running StarGROUP. If a host is also arranged to provide file serving then those disk drive letters are also available to Kermit for normal DOS file operations. Most of the testing of C-Kermit 5A with MS-DOS Kermit 3.0 at Utah State University occurs across a StarLAN local area network.

DECnet-DOS, or DECnet-PCSA, is the PC end-node version of Digital Equipment Corporation's DECnet networking method used to form networks, of local to worldwide dimensions. Among the features of DECnet-DOS is a terminal emulator named SETHOST, named after the VAX command SET HOST used to establish a direct connection to any machine on the network. MS-DOS Kermit replaces that program with itself. Once DECnet-DOS main-body software is running on the PC and either CTERM or LAT have been loaded give the MS-DOS Kermit command SET PORT DECNET *nodename*, CONNECT, and see a login prompt from the distant host.

DECnet-DOS provides two pathways for terminal connections: CTERM (Command Terminal) and LAT (Local Area Transport). CTERM works to any VAX near or far, LAT is for connections on the local Ethernet; load one program or the other depending on distance or what the local site requires. MS-DOS Kermit will transparently try LAT first (it's faster) and then if necessary the CTERM path. HANGUP and BREAK (`\Kbreak`, Alt-B) work here too.

DECnet-DOS has many other features, among which is invocation of a named task on another DECnet-DOS node temporarily acting as a server. MS-DOS Kermit can be such a task —a Kermit server upon demand. Details are part of DECnet SPAWNER (guru territory).

Systems managers enjoy two additional features. SET PORT DECNET * shows a list of all the LAT servers on the network. The SET PORT DECNET command can also have an extra word for the password of a LAT box: SET PORT DECNET *node password*, for management and security purposes.

DEC PCSA users will be happy to know that MS-DOS Kermit also runs well under Microsoft Windows, in a window like other Windows programs. Windows for AT (80286) machines imposes one constraint: only the active window receives service for serial port communications (COM1 etc); the 386 version lifts this restriction. Also Windows for 386 machines permits Kermit to do full color graphics but Windows 286 will not let Kermit do graphics in a Window, but will allow full-screen access if Kermit is marked as ''Exclusive'' for graphics.

TCP/IP is a very widespread networking method linking machines locally and around the world. Kermit does not attempt to speak TCP-ese directly, because that requires a large specialized body of code in itself. But Kermit can be used as the terminal in Telnet connections for some TCP/IP products. The way it works is to tell Kermit SET PORT BIOS1, CONNECT. The TCP/IP product must provide a connection between that apparent serial port point, BIOS1, and its own code to send and receive characters across the network. The general name for this connection is an Interrupt (INT) 14h interceptor. On a PC thus equipped, MS-DOS Kermit can replace the normal Telnet program, and it can also transfer files over the same connection.

Presently we know of three commercial products supporting this connection: FTP Software Inc.'s TNGLASS routine running with their kernel software, Interlan's TCP/IP Gateway for Novell Networks (described above), and Novell's Excelan LAN WorkPlace for DOS. The latter provides INT 14h service, 3COM BAPI, and other interfaces usable by MS-DOS Kermit 3.0. It is hoped that public domain (or at least free) TCP/IP packages such as NCSA Telnet will also add INT 14h service that Kermit can take advantage of. In the commercial products a small interface program, such as TNGLASS, is given the name of the remote host and it starts Kermit when the host responds, for example:

```
tnglass hostname -e kermit commands
```

Then the Kermit commands begin a terminal session with the host. EXITing Kermit normally ends the session; just ''shelling out to DOS'' (Kermit's PUSH or RUN commands) keeps it alive.

Summary

With all its new communication features, MS-DOS Kermit is rapidly becoming a ''total solution'' to campus and corporate communication needs. Rather than having to use one program for dialing, another for Telnet, another for local mainframe communication, yet another for accessing asynchronous communication servers, and still another for station to station functions, the PC user can do it all with MS-DOS Kermit 3.0. This brings substantial savings in software costs, disk storage, and perhaps most important, in ''human factors'' by reducing the time required to learn multiple packages and the inconvenience of frequently switching among them.

MS-DOS Kermit 3.0 In Print

Frank da Cruz
Columbia University, New York City, USA

Christine Gianone's book *Using MS-DOS Kermit* is a milestone in Kermit documentation. Unlike our usual pile of computer printout, it is a professionally produced, typeset, illustrated work, lavished with useful examples and tables, gently written for the ''mere mortal'' who winces at data communications jargon, interface specifications, and circuit diagrams and who merely wants to set up and use communication software with a minimum of pain and fuss. In the words of Joe Doupnik:

> ''*Using MS-DOS Kermit* covers all the topics needed to start new users easily, and it does so in a natural progression . . . The pace is a lot faster than readers will realize, mostly because the writing is skillfully done − crisp, often entertaining, and always to the point. This is a terrific book ! ''

And not just for new users − the reference sections of my copy are already well-thumbed and rumpled! The appendices listing the IBM PC's keyboard scan codes, keyboard verbs, and character sets are invaluable, and the chapter on International Character Sets is a groundbreaker.

I expect that Chris's book to go a long way towards ''legitimizing'' Kermit as a real software product on the mass market, and introducing it to a large sector of the public that might not have heard about it before. And don't worry, this publication does not signal any change in the status of Kermit software. You can still copy and share it (the software, not the book !). Here are some early reviews from readers:

> ''*Using MS-DOS Kermit* is very well written and clarifies many questions that I've had for several years. I love Kermit because . . . it's command-line oriented, because it talks to practically all machines, and most important because it's the product of a community of dedicated people instead of what one finds increasingly in the computer world.'' − *Norman Miller, Trinity College*

> ''Congratulations and thank you for a job well done. I have been using Kermit for my connection to the campus mainframe for several years and wish that your book had been available when I first started out. Until now I have been using V2.32/A but with the new bells and whistles of V3.0 (and many of the old ones that you have brought to my attention) I shall be able to use Kermit even more effectively. I have recommended to our PC support group that they promote your book for first-time users of Kermit as it is so very readable and contains so many practical examples to clarify the commands.'' −*Trevor Cradduck, Univ. of Western Ontario*

> ''[*Using MS-DOS Kermit*] is truly worthy of the beautiful foreword and could serve as a pradigm for thoughtful technical writing.'' − *Hal Falk, City University of New York*

> ''I just received Christine's book from Digital Press and am very pleased with the style and clarity with which it is written. I have installed older versions on 30-40 PCs and am totally happy with the package. I think it's great and you all are to be congratulated for your efforts.'' − *John F. Waters, National Cancer Institute*

Using MS-DOS Kermit will be available soon in ''soft copy'' to people with print disabilities from Computerized Books for the Blind, 52 Corbin Hall, University of Montana, Missoula, Montana 59812, USA, Telephone (406) 243-2899.

Making the Mainstream Connection with MS-DOS Kermit

Robert J. Arnzen, DSc., St. Louis, Missouri, USA

The old adage ''Man is a tool using animal'' has been with us for quite some time and the image that commonly springs to mind is that of a person wielding a hand tool or using a machine to multiply muscle power or perform a task that the human anatomy is otherwise ill equipped to accomplish. With today's widespread availability of information processing ''tools'' it may be much more appropriate to revise the old familiar adage and state that ''Man is a symbol using animal'' and the contemporary image might be one of a person seated before a personal computer, busily typing on a keyboard and viewing a video screen filled with softly glowing symbols. The scene is commonplace and the range of possibilities that exist for symbolic manipulation, that is to say, information processing has become enormous. Moreover, when large numbers of these systems are interconnected, the scope of application possibilities become astronomical. The potential personal benefits to be gained are correspondingly great.

Unfortunately, past experience has shown that persons with severe physical handicaps of one form or another have not been able to enjoy the full or even partial benefits that many new technologies provide and this has been especially true for the visually disabled community. Happily, the advent of the personal computer revolution did not follow that path. As personal computer systems emerged in large numbers in the mid 70's, the proliferation stimulated a host of new products for these systems. Of particular importance to the potential blind users of PCs was the introduction of a number of speech synthesizers in the early 1980's that were easily adapted to personal computer hardware. The devices were not specifically targeted to the blind user but, rather, manufacturers had the general population of PC owners in mind as a market base. Consequently, the resultant high volume production of these units reduced their cost dramatically and afforded the visually disabled an opportunity to tap into the wealth of information that was rapidly becoming widely available.

One of the principal ingredients of a speech equipped PC is the mechanism by which new information is detected and captured on its way to the video display system and subsequently processed for use by the speech synthesizer. Several methods have been employed but the one that is frequently favored for use with the MS-DOS operating systems running on IBM compatible equipment utilizes the Terminate and Stay Resident (TSR) DOS function call or the DOS device driver facility to install a program that remains resident in the memory of the machine after loading is completed. In a nutshell, the function of this program is threefold. First, it must detect and trap any new information going to the video display system and the data is commonly copied into a private buffer area so as not to adversely impact program execution speed. Second, the program must monitor the keyboard for any special instructions to change mode or to review the information displayed on the video screen. Finally, the program must send the information it collects and processes to a speech synthesizer attached to the PC. Of course, all of this must happen in a manner that is completely transparent to the operation of a running program or to the user. Clearly, these requirements can be extremely difficult to satisfy and they become especially difficult in the case of communication programs that are necessarily time critical processes.

In light of the stringent demands placed upon a communication program running in a speech synthesized environment, Kermit has proven to be a marvelous file transfer and terminal emulation system that I have found extremely useful for my work with personal computers over the past 6 years. As an individual with a visual handicap, Kermit has consistently proven to function in harmony with the speech synthesis programs upon which I am completely dependent for gaining access to the visual information displayed on a video monitor. Kermit is a well behaved, robust protocol that can function reliably in a hostile environment and has provided the blind community a valuable tool for making the mainstream connection to a fabulous source of information.

Editorial Note — Kermit developers have always made a special effort to keep Kermit programs usable by people with visual, hearing, or motor impairments. Other communication packages that fill the screen with brightly colored menus, graphics and sound effects, and whose functions are invoked by arcane key combinations like Ctrl-Alt-Shift-F10, are not compatible with most speech and other prosthetic devices. As one disabled user (and developer) of Kermit puts it,

> ''In about 15 milliseconds I went from a 120 wpm touch typist to a 35 wpm two-'finger' typist. Kermit's command abbreviation and completion feature is a big plus for higher quadruplegics and cerebral palsy types who use special keyboards. For some of those folks, a keystroke is a 10- to 120-second ordeal.'' — *Warren Tucker, Mountain Park, Georgia, USA*

C-Kermit 5A Is Coming . . .

C-Kermit 5A for UNIX has reached the testing stage, and it will be released Real Soon Now. Written by Frank da Cruz of Columbia University, with contributions from many others around the world, this version includes:

- Performance improvements. Sliding window packet protocol, which may be used with both short and long packets. Maximum packet length increased to 2000 (maybe more by the time the program is released). Low-level packet readers and writers recoded for increased efficiency. Dynamic sizing of packets based on error history. See the article on page 12 for a discussion of these features.
- Transfer of text files in most Roman-alphabet-based languages using the new Kermit international text transfer protocol (see article, p.16). C-Kermit also includes special language-specific transliteration rules for rendering languages like German, Norwegian, and Italian into US ASCII, which lacks accented characters.
- Sending and recognition of file attribute (A) packets: file size, creation date, file type, character set, system ID, as described in *Kermit News* V3 #1, June 1988. Control of file attributes on an individual basis.
- A new and powerful script programming language compatible with that of MS-DOS Kermit, including macros and parameters, INPUT and OUTPUT, IF and GOTO, etc, but with many extensions including arithmetic and string functions, a local file input/output and management system, user-defined functions, local and global variables, arrays, access to host functions and environment variables, FOR and WHILE loops and IF-ELSE constructions with nesting and statement grouping, and much more.
- On Berkeley-based UNIX systems, support for TCP/IP telnet connections. Kermit replaces TELNET and FTP! Now you can have a script language for use over TCP/IP connections, and for the first time you can transfer ''national language'' files between unlike computers in the TCP/IP environment.
- A new set of FILE COLLISION options − APPEND, BACKUP, DISCARD, OVERWRITE, RENAME, UPDATE − that specify what to do when a file arrives that has the same name as an existing file. For example, UPDATE means to replace the existing file only if it is older than the arriving file.
- Security features for server mode: DISABLE and ENABLE commands similar to MS-DOS Kermit.
- Non-error-checked uploading and downloading of files from computers that don't have Kermit.
- Improved and expanded modem control, and many, many other new features.

C-Kermit 5A has been built and tested on most post-V7 varieties of UNIX, including AT&T System V R2 and R3, Berkeley 4.1, 4.2, and 4.3, most varieties of Xenix, DEC Ultrix, SUN OS, Encore UMAX, NeXT Mach, HP-UX, IBM AIX, Masscomp RTU, and DIAB DNIX. Because of all its new features, version 5A is larger than previous releases and might not run on computers with small address spaces; older releases of C-Kermit can be used on small systems. Adaptation of C-Kermit 5A to VAX/VMS, IBM OS/2, the Apple Macintosh, and the Commodore Amiga is underway.

Special thanks are due to the volunteer programmers who have put so much effort into development and testing of C-Kermit 5A in diverse environments and locales: Kristoffer Eriksson (Öerobro, Sweden); Bo Kullmar (Stockholm, Sweden); Warren H. Tucker (Mountain Park, Georgia, USA); Peter Mauzey (Middletown, New Jersey, USA); Joe R. Doupnik (Utah State University, USA); Ken Yap (University of Rochester, New York, USA); Paul Placeway (Cambridge, Massachusetts, USA); Chris Adie (Edinburgh University, Scotland, UK); Chris Armstrong (New York, USA); Mark Buda (New Hampshire, USA); Steve Walton (California State University Northridge, USA).

Contact Kermit Distribution at Columbia for availability of C-Kermit 5A. The UNIX version should be ready, at least in beta test form, by early Summer 1990, with the VMS, Macintosh, OS/2, and other versions to follow later.

Macintosh Kermit

The new release of Macintosh Kermit, when it is ready, will not only have the new protocol features of C-Kermit 5A, but possibly also a way to get at the new script language. In addition, Paul Placeway (the Macintosh Kermit developer) has been adding features like VT320 emulation with international characters, a sizeable terminal window with scrollback, MacBinary transfers, printer support, and improved keyboard management.

IBM Mainframe Kermit 4.2 Is Here

John Chandler
Harvard/Smithsonian Center for Astrophysics
Cambridge, Massachusetts, USA

There have been two major releases of IBM Mainframe Kermit-370 since the last issue of *Kermit News*. This program is also known as ''Portable Kermit-370'' because it is designed for easy adaptation to any operating system that runs on an IBM System/370-architecture machine, including the entire IBM 370 line (43xx, 303x, 308x, 3090, 937x), plus Amdahls and other compatibles. To date, the Kermit-370 program has been successfully introduced to three major software environments: VM/CMS, MVS/TSO, and MUSIC/SP.

Kermit-370 version 4.1 was released in January 1989 for VM/CMS and MVS/TSO. It included many bug fixes, file transfer protocol improvements, new commands, and, perhaps most important, support for additional types of communication front ends (see below). The following month, Pierre Goyette of McGill University completed adapting this program to McGill's MUSIC timesharing system.

Version 4.2 of Kermit-370 was released in May 1990. Its major features include an accounting function, a method of sending partial files by line numbers, improved support for Attribute packets, support for the new file collision mechanism, and support for a *large* selection of international character sets for file transfer (see p.16).

As yet, Kermit/370 does not support CICS or DOS/VSE. However, among the Kermit-370 files is support code for a version of CICS that runs in a rather specialized environment in the Soviet Union (see article on p.18). We are hoping this CICS support will be merged into Kermit-370 soon. A CICS version should be usable with CICS under any IBM/370 operating system, including MVS, VM, and DOS/VSE. Efforts are currently underway at Lehigh University, but further contributers and testers are always welcome. Volunteers? Please contact us if you're interested.

The IBM Difference

Because IBM mainframes use a different character set from most other computers, and because their preferred method of terminal communication is full-screen block-mode, it can be quite a trick to get Kermit file transfer to work between an ordinary ASCII computer and an IBM mainframe. To complicate matters, there is a seemingly endless parade of different front ends, controllers, concentrators, and simulators to choose from, each different from the other.

One of Kermit's claims to fame is that, unlike most other asynchronous protocols, it can work at all in this environment. Two methods are offered. The first method requires a ''linemode'' or TTY connection to the mainframe, in which the front end converts incoming ASCII to EBCDIC, and then Kermit converts the EBCDIC back to ASCII in order to process the packet control fields (length, checksum, etc), and then back to EBCDIC again to store the data on disk, plus the reverse process for sending data. This works even when special character sets like Hebrew, Greek, or Cyrillic are involved because Kermit's packet control fields are the simple, printable ASCII characters that are available in all character sets. The data within the packets is handled separately.

The second method is used through special front ends called ''protocol converters'', which not only perform character code conversion, but also manage the screen of an ASCII terminal (or emulator, like MS-DOS Kermit) to give the illusion of an IBM full-screen 3270 block-mode terminal. The protocol converter's behavior is unpredictable; it might, for instance, decide not to transmit certain characters because they are unchanged since the last time the ''screen'' was updated −an undesirable feature when files are being transferred! To transfer files successfully through a protocol converter, Kermit-370 must turn off protocol conversion by putting the converter into ''transparent mode''. Unfortunately, each protocol converter seems to have a different way of doing this, and some have no way at all. And to compound matters further, front ends can be connected in series!

Therefore, we cannot guarantee that Kermit-370 can work in all communication environments. We do know, however, that it works in TTY linemode with the IBM 3705, 3708, 3725, and the COMTEN 36xx series, and that it works in full-screen mode with the IBM Series/1, 4994, and 7171 front ends and with the IBM 937x ASCII subsystem, and with compatible protocol converters including the Hydra II and Commtex Cx-80. It even works through an IBM 8232 with TCP/IP from another computer running TN3270. In many configurations, Kermit-370 can recognize front ends automatically by issuing a special order. Front ends that do not have the Series/1-style transparent mode might still permit Kermit file transfers in ''graphics'' mode. Kermit-370 makes a ''best guess'' as to the correct mode when it starts up. Graphics mode works on the Datastream/Leedata 8010, 8030, and 974, the PCI 1076 and 276, the Renex TMS-1 and RTD, and with KMW front ends.

Failure has been reported with Adacom products, the MICOM 7400, Hydra/SNA, Datalynx 3274, and with the SIM3278 software protocol emulator. The IBM 3174 controller does not have a transparent mode suitable for Kermit file transfers, but reportedly IBM will add this capability in a forthcoming microcode upgrade. For others, the results are not yet in. Reports welcome! IBM Mainframe Kermit-370 is available on *Tape B* (see the order form in back).

New Kermits for DEC Computers

VAX/VMS Kermit-32, long considered a dead product because the original authors, Nick Bush and Bob McQueen of Stevens Institute of Technology in Hoboken, New Jersey, USA, had moved on to new jobs, received a welcome enhancement from Burt Johnson of Diversified Computer Systems, Boulder, Colorado, USA, in February 1990: support for long packets (see p.12), up to 1000 characters in length. This update is an adaptation of Dan Norstedt's Kermit-10 release for the DECsystem-10 (see below). Jonathan Welch of the University of Massachusetts, who had fixed some bugs in earlier versions, added several new features to Burt's submission, most notably the long sought-after SET FILE BLOCKSIZE command for receiving binary files, and also allowance for file names and types up to 39 characters. The result is version 3.3.122, released in May 1990, now available on Tape B (see order form after page 30).

There is not likely to be much further large-scale development of Kermit-32, mainly because it is written in a language, *Bliss*, that very few sites have compilers for. Work is in progress on adapting C-Kermit 5A to VAX/VMS, and when that's done we'll have a more powerful and supportable Kermit program for VAX/VMS. In the meantime, thanks to Burt for this much-needed performance boost to Kermit-32, and to Jonathan for his welcome contributions.

PDP-11 Kermit-11 for RSX-11, RT-11, RSTS/E, TSX+, P/OS, Pro/RT, and IAS had a minor release (3.60) in June 1989 from its creator, Brian Nelson at the University of Toledo, Ohio, USA. This release includes bug fixes, additional file attribute packet capabilities, improved CONNECT mode for RSX, and improved support of the DF224 modem. *Tape B*.

DECsystem-10 Kermit-10 version 3(134) was contributed by Dan Norstedt of the Stacken Computer Club in Sweden in September 1989. Its major feature is support for long packets, up to 1000 characters, plus several bug fixes. Since Kermit-10 and VAX/VMS Kermit-32 share a common Bliss-language protocol module, Dan's work was quickly adapted to Kermit-32 (see above). *Tape B*.

PDP-8, PDP-12, and DECmate were brought into the fold in October 1989 with Kermit-12 from Charles Lasner. This program runs on the entire series of DEC 12-bit computers including the entire PDP-8 and DECmate lines under any OS/8 family operating system, including compatibles like the Fabritek MP-12, Intersil Intercept, Pacific CyberMetrix PCM-12, DCC-112, Computer Extensions boards, the Hungarian TPA, etc. Kermit-12 also runs on the DEC PDP-12 laboratory computer (circa 1970) and on the DEC PDP-10, PDP-15, or IBM PC configured with a suitable PDP-8 emulator. *Tape D*.

Ultrix. *Tape B* (see C-Kermit article).

New Hewlett-Packard Kermits

HP-1000. Version 1.99D of HP-1000 Kermit for the Hewlett-Packard 1000 series of computers with the RTE-6 and RTE-A operating systems was contributed by Paul Schumann of E-Systems, Inc, Greenville, Texas, USA, in January 1990. This version replaces version 1.98 of September 1986. The major new feature is support for the HP D-Series multiplexer and drivers. *Tape C*.

HP-9000 BASIC Workstation. This is a new Kermit program, written entirely in HP BASIC, contributed by Andy Campagnola of the Hewlett-Packard Measurement Systems Division in Loveland, Colorado, USA, with contributions from Keith Moore at the HP New Jersey Division labs. Kermit-RMB is capable of both remote and local operation, transfers text and binary files over both 8-bit and 7-bit connections, emulates the DEC VT100 terminal, does raw uploads and downloads, includes logging functions, has a macro facility, and handles Kermit File Attribute packets. It also includes printer control, a hex editor and dump facility, and a file type conversion utility. The program runs on any Series 200/300 computer with 1MB of memory and BASIC 4.0 or greater, and will run on any HP serial interface available for these computers. *Tape C*.

HP-125 Business Assistant with CP/M 2.2. An adaptation of CP/M Kermit 4.09 from Mike Freeman, Bonneville Power Administration, Vancouver, Washington, USA. Aug 89. *Tape A*.

HP MS-DOS PCs. Versions 2.31, 2.32, and/or 3.0 for the HP-110, 150, and Portable PC. *Tape A* (see MS-DOS Kermit article).

HP-UX. *Tape B* (see C-Kermit article).

Other New Kermits

Since the last issue of *Kermit News:*

PRIME Kermit version 8.00 was contributed in January 1990 by John Horne, Polytechnic South West (PSW), Plymouth, Devon, England, UK. The new version replaces version 7.57 of May 1986, which was contributed by The Source Telecomputing, Alexandria, VA. The Source was recently bought up and liquidated by Compuserve. Thanks and goodbye to our friends at The Source who contributed so much to Kermit in the early days!

The new version has been tested at PRIMOS revisions 21.0.5q and 22.0.1a. Changes include: support for 8-bit no-parity file transfers; better error handling and messages; full support for pathnames within commands; improved logging; more command line options available; pound sign conversion option (US/UK); support for file size and date attributes; support for nested TAKE files; local file management commands added; filename collision detection and avoidance.

Version 8.11 is expected soon. It will include improvements in the sliding window algorithms (so that it works nicely with the new MS-DOS and C Kermits), support for dialout lines and local-mode operation (CONNECT, GET, FINISH, BYE), and script commands (INPUT, OUTPUT, PAUSE, and CLEAR). This work, done by Matthew Sutter of Lincoln National Corporation, Fort Wayne, Indiana, USA, has just been sent to John Horne at PSW for final checkout before release. Contact us at Kermit Distribution for availability. *Tape D*.

Honeywell DPS-6 Kermit comes from Frank Dreano, Chesapeake, Virginia, USA (who also wrote the article on p.21). Its features include file attribute packets and long packets. Documentation and C source code are included. Due to peculiarities of the Honeywell environment, a modified version of MS-DOS Kermit is required to talk to it; this is also supplied. Support for Honeywell communications is expected to appear in a future release of ‘‘main-line’’ MS-DOS Kermit and perhaps in other Kermits as well. *Tape D*.

Apollo Aegis Pascal Kermit 2.9, handles 8th-bit prefixing and repeat-count compression, improved CONNECT mode. From Gordon Sands, Marconi Space Systems, Portsmouth, England. May 89. *Tape C*.

Apple II Kermit 3.86, Oct 89, for Apple DOS and ProDOS, from Ted Medin, Naval Ocean Systems Center. Improved performance, bug fixes, multiple protocols supported, new commands TYPE, TAKE, HELP. *Tape A*.

BTOS/CTOS Kermit 1.07 for Burroughs B20, Convergent NGEN: Joel Dunn, Univerity of North Carolia at Chapel Hill, USA. July 88. *Tape C*.

GEC OS4000 Kermit 3.9, bug fixes done by Gordon Sands, Marconi Space Systems, Portsmouth, England. May 89. *Tape D*.

IBM PS/2 OS/2 Kermit 1.00, based on C-Kermit 4E(072), from Chris Adie, Edinburgh University Computing Services, Scotland, UK, May 89. (A newer version, based on C-Kermit 5A, is in preparation.) *Tape B*.

Luxor ABC Series. ABC-800, ABC-80. From Bo Kullmar, ABC-Klubben, Stockholm, Sweden. Dec 89. *Tape C*.

Microsoft Windows Kermit 4.11 from Bill Hall. H19 terminal emulation, Kermit file transfer under Microsoft Windows, with a Windows-style user interface. Jan 90. *Tape A*.

NEC PC 9801 Kermit 2.32/A with Kana/Kanji character support, by Hirofumi Fujii, National Laboratory for High-Energy Physics, Japan, Apr 89. *Tape A*.

PICK Kermit version 0.3 from Joe Fisher of Austin, Texas, USA, for McDonnell Douglas (MicroData) REALITY 4.2E, DEC MicroVAX II with Ultimate Coprocessor, IBM PC/XT and PC/AT with PICK R93. Aug 89. *Tape D*.

Uniflex 6809 Kermit from Jur van der Burg, Alphen aan den Rijn, Netherlands, v1.4, Jan 89. *Tape C*.

How Efficient Is Kermit?

Frank da Cruz and Christine Gianone

When Kermit was born nine years ago, file transfer efficiency was close to the bottom of the priority list. An efficient design was definitely at odds with the primary goals of universality and simplicity. In order to make Kermit work through delicate front ends and cantankerous communication devices, its packets were deliberately kept short, with all data encoded as printable ASCII characters. And for Kermit to work in the IBM mainframe environment, the protocol had to be half-duplex ‘‘stop and wait’’.

These design decisions gave Kermit easy entrée into territory unexplored by other popular file transfer protocols, and largely account for its popularity. But now that Kermit works reliably on a wide range of computers, there is an increasing demand to make it also work more efficiently.

In this article, we'll look at Kermit's performance, see how it has been improved by protocol extensions, and measure the performance of new Kermit releases. This discussion requires some knowledge of the Kermit protocol (as described in the book *Kermit, A File Transfer Protocol*, Digital Press, 1987). We begin with the basic Kermit packet:

Ctrl-A	LEN	SEQ	TYPE	DATA . . .	CHECK

The length field, LEN, is a single printable 7-bit ASCII character that tells the length of the rest of the packet. Since there are 95 printable ASCII characters, the maximum length is $95 - 1 = 94$. SEQ and TYPE are single-character fields, so the longest possible data field is 91.

For an asynchronous serial connection, we define *efficiency* as:

$$e = \frac{f/t}{r/10} \qquad (1)$$

where:

- e = efficiency, 1.00 is perfection
- f = file size in characters
- t = elapsed time in seconds
- r = transmission rate, bits per second

This is the ratio of actual file characters transferred per second to the speed of the communication line in characters per second (cps). The transmission speed r is divided by 10 to convert bits per second (bps) to cps because a character is transmitted as 8 bits framed by 1 start bit and 1 stop bit, 10 bits in all.

The basic Kermit protocol works by sending packets back and forth in stop-and-wait fashion. The file sender transmits a packet containing file data in its DATA field, the receiver sends back an acknowledgement packet (ACK) with an empty DATA field. Let's assume the connection between the two Kermits is perfectly clean, with no transmission delay, and that the two computers take no time at all to compose and process packets. Formula (2) shows the expected efficiency for a text file transfer with the basic Kermit protocol. It is the ratio of the number of data characters in a packet to the total number of characters required for the packet to be sent and acknowledged.

$$e = \frac{0.95 \times p}{p + 10} \qquad (2)$$

where:

- e = efficiency, 1.00 is perfect
- p = packet data field length
- 0.95 = encoding factor
- 10 = protocol overhead characters

The *encoding factor* results from the ‘‘single shift’’ character, or prefix, that precedes each control character in the data field of a Kermit packet for transparency. The average text file contains about 5% control characters, so 5% of a typical DATA field is overhead (the figure is higher for binary files). The *protocol overhead* consists of the five control fields in the data packet plus the five control fields in the dataless acknowledgement. Table 1 shows how efficiency increases with packet length, and that the *best* efficiency we can expect from the basic Kermit protocol is 85.6% for typical 7-bit text files.

Table 1 *Basic Kermit Efficiency*

Data Length	Efficiency
20	0.633
40	0.760
80	0.844
91	0.856

But now suppose that data does not travel between the two computers instantaneously. It might have to travel very long distances or stop and visit a number of intermediate devices like packet-switch nodes in a public data network. How does Kermit fare under these conditions? The formula is:

$$e = \frac{0.95 \times p}{p + 10 + c} \qquad (3)$$

where:

- p = packet data field length

$$c = d \times \frac{r}{10}$$

d = round trip delay, seconds
r = transmission rate, bits per second

c is the number of characters that can be transmitted at r bits per second in d seconds, that is, the delay expressed in character times. Formula (4) shows Kermit's efficiency for $p = 91$ and $r = 1200$:

$$e = \frac{91 \times 0.95}{91 + 10 + (d \times 120)} = \frac{86.45}{101 + (d \times 120)} \qquad (4)$$

The efficiencies for delays from 0 to 5 seconds from formula (4) are given in Table 2. As you can see,

Table 2 *Basic Kermit Efficiency with Delays*

Delay	Efficiency
0	0.856
1	0.391
2	0.254
4	0.149
8	0.082
16	0.043

efficiency deteriorates rapidly as delay increases. For 1200 bps, efficiency is cut about in half as delay doubles, and the effect is even more pronounced at higher transmission rates.

Long Packets

An easy and effective way to improve Kermit's efficiency is to use longer packets to increase the ratio of real data characters to packet overhead characters and decrease the number acknowledgements required. An alternative ''extended packet'' format, designed in 1985 and implemented since then in most popular Kermit programs, sets the normal LEN field to an otherwise illegal value of zero (space), and adds three new fields —a two-byte length field and a one-byte header checksum —at the beginning of the DATA field, to allow lengths up to $95^2 - 1 = 9024$. Replies (ACKs) are still short packets. The efficiency formula for a text file transfer with long packets is:

$$e = \frac{0.95 \times p}{p + 13} \qquad (5)$$

The protocol overhead in this case is 13, rather than 10 as in formula (2), the sum of the 8 control bytes in the extended data packet and 5 in the ACK. Table 3

Table 3 *Long Packet Efficiency*

Data Length	Efficiency
100	0.840
200	0.892
500	0.926
1000	0.938
5000	0.948
9024	0.949

shows the efficiencies given by formula (5) for different extended packet lengths on a clean connection with no delays. When transmission delays are considered, the long packet efficiency formula becomes:

$$e = \frac{0.95 \times p}{p + 13 + c} \qquad (6)$$

which is just like (3), but with 13 protocol overhead characters rather than 10. Table 4 shows the efficiencies for different packet length and delay values at speed r=1200.

Table 4 *Long Packet Efficiency with Delays*

	Packet Length			
Delay	91	500	1000	9024
0	0.856	0.926	0.938	0.949
1	0.391	0.750	0.838	0.936
2	0.254	0.631	0.758	0.924
3	0.188	0.544	0.692	0.912
4	0.149	0.478	0.636	0.901
5	0.123	0.427	0.589	0.890

So for a given delay, efficiency increases with packet length. Or does it? On a noisy connection, longer packets are more likely to be damaged by interference and take more time to retransmit; under these conditions, long packets can actually *reduce* efficiency.

Dynamic Packet Length

To cope better with noise, the file sender can vary the packet length based upon the error history. When a bad checksum is detected, the packet length is reduced; as packets arrive with good checksums, the packet length can be gradually increased back up to the negotiated maximum. The best length at any given time is a function of different factors, and can be calculated using an algorithm presented by John Chandler in the article ''Dynamic Packet Size Control,'' in *Kermit News* V3 #1, June 1988. Presently, IBM Mainframe Kermit (version 4.0 and later) and C-Kermit 5A incorporate this technique, which can be used equally well on both full- and half-duplex connections.

Sliding Windows

A technique called ''sliding windows'' addresses both noise and delay. Sliding window protocol, available in MS-DOS Kermit 3.0, C-Kermit 5A, and in PRIME Kermit, does not stop and wait for a reply after transmitting a packet, but rather transmits packets continuously and permits the acknowledgements to come later. Packets literally pass each other on the communication line, like cars on a divided highway. A full-duplex connection (two-way street) is required.

The number of packets that can be sent before acknowledgements arrive is called the ''window size''. Whenever the oldest packet in the window is acknowledged, the window can ''slide'' forward and a new packet can be sent. If a packet is damaged by noise, only that packet need be resent. Therefore, with a relatively small packet size, efficiency can be high even under noisy conditions.

Kermit's window size can be 1 to 31. What is the best window size to use? A good Kermit implementation does not preclude mixing long packets and sliding windows so it is conceivable to have 31 window slots each with a 9024-character packet, requiring 274K of memory. In practice, programs like C-Kermit and MS-DOS Kermit offer only about 4-6K for packet buffers due to memory, addressing, and other limitations. To allow for longer packets, we should use the smallest adequate window size. The formula for the minimum window size w to achieve no delay is:

$$w = \frac{d \times \frac{r}{10}}{L} \qquad (7)$$

where:

d = round-trip delay, seconds
r = transmission speed, bps
L = total packet length

If w comes out to zero, then we use 1. When using an adequate window size, transmission of data can be continuous − a phenomenon best appreciated by watching your modem's receive and transmit lights. Table 5 shows minimum window size for various speeds and delays with a 91-character data field.

Table 5 *Window Sizes for p = 91*

bps	Delay (sec) 0	1	2	3	4	5
9600	1	11	21	*32*	*42*	*52*
4800	1	6	11	16	21	27
2400	1	3	6	8	11	14
1200	1	2	3	4	6	7
300	1	1	1	1	2	2

As you can see, at high speeds with long delays we need a bigger window than Kermit has. This is where long packets and sliding windows complement each other. Table 6 shows the minimum window sizes for 200-character packets.

Table 6 *Window Sizes for p = 200*

bps	Delay (sec) 0	1	2	3	4	5
9600	1	4	9	14	19	24
4800	1	2	4	7	9	12
2400	1	1	2	3	4	6
1200	1	1	1	1	2	3
300	1	1	1	1	1	1

With an adequate window size and therefore continuous transmission, neither delays nor ACKs contribute to the total transmission time, as shown by formula (8) for regular packets:

$$e = \frac{0.95 \times p}{p + 5} \qquad (8)$$

and formula (9) for long packets:

$$e = \frac{0.95 \times p}{p + 8} \qquad (9)$$

Table 7 shows the theoretical maximum efficiency for different packet lengths resulting from these formulas.

Table 7 *Efficiency with Sliding Windows*

Packet length	Efficiency
91	0.901
500	0.935
1000	0.945
2000	0.946
9024	0.949

Data Compression

An additional performance boost comes from a simple kind of data compression in which repeated bytes are compressed into a special prefix, a count, and one copy of the repeated byte. Analysis of many megabytes of textual and binary information shows an average 15-20% reduction in transmitted characters when this technique is used (see *Kermit, a File Transfer Protocol*, pp.248-251). Most popular Kermit programs negotiate this kind of data compression with each other automatically. The major beneficiaries are text files that contain sequences of blanks within indented or columnar material and binary files containing initialized (zero) data. For example, the MS-DOS 3.30 program `FIND.EXE`, 6417 bytes long, can be sent by Kermit in a single 780-byte packet (efficiency = 817.5%).

Benchmarks and Bottlenecks

Now let's compare theory with reality. Table 8 presents actual efficiency measurements, based on formula (1), for transferring a 36K text file containing no repeated characters between MS-DOS Kermit 3.0 on a PS/2 Model 50 with a hard disk and C-Kermit 5A on an unloaded SUN-4 through a direct terminal connection at 2400 bps (no delay). Up means uploading from the PC to the SUN, Dn means downloading from the SUN to the PC.

Table 8 *PS/2 - SUN Benchmark, 2400 bps*

Packet Length	Window Size 1 *Up*	1 *Dn*	2 *Up*	2 *Dn*	4 *Up*	4 *Dn*
90	.74	.73	.86	.88	.86	.88
250	.84	.84	.88	.91	.89	.89
500	.88	.88	.91	.91	.89	.91
750	.89	.88	.91	.91		
1000	.91	.89	.89	.93		
1500	.89	.89				
2000	.91	.91				

Table 9 shows Kermit transfers on the same computers and connection, but at 9600 bps. Comparing the two cases reveals efficiency to be lower at higher speeds (but of course the actual throughput is still higher). This suggests that there is some transmission speed at which one of the computers becomes the bottleneck. To find out which one, we perform the same benchmark again, this time between two SUN-4

Table 9 *PS/2 - SUN Benchmark, 9600 bps*

Packet Length	Window Size 1 Up	Dn	2 Up	Dn	4 Up	Dn
90	.65	.64	.79	.87	.80	.87
250	.76	.76	.82	.91	.84	.91
500	.79	.80	.87	.91	.88	.93
750	.84	.82	.87	.93		
1000	.84	.82	.87	.93		
1500	.85	.83				
2000	.85	.84				

computers running C-Kermit 5A at 9600 bps, with the results shown in Table 10.

Table 10 *SUN - SUN Benchmark, 9600 bps*

Packet Length	Window Size 1 Up	Dn	2 Up	Dn	4 Up	Dn
90	.73	.73	.89	.89	.89	.89
250	.81	.81	.91	.91	.91	.91
500	.87	.87	.93	.93	.91	.91
750	.89	.89	.93	.93		
1000	.87	.87	.96	.96		
1500	.89	.89				
2000	.91	.91				

These results are not surprising, because the SUN is a much faster computer than the PS/2. For a given computer (ignoring load), the packet processing and disk access time remain constant, and therefore begin to dominate the efficiency as transmission speed increases. Comparing the figures in Table 10 with the theoretical limits for each case shows that C-Kermit 5A on the SUN approaches (and in one case exceeds) theory when the window size is greater than one, but falls somewhat short in the nonwindowed case where packet processing and disk accesses occur between packets.

Our final set of measurements, made by Warren Tucker of Tridom Corporation, Marietta, Georgia, USA, shows the effects of delays induced by a long-distance connection, in this case through the AT&T Tridom Clearlink satellite communications network. A 20K text file was sent using varying packet and window sizes at 9600 bps between a Pyramid 90/x running C-Kermit 5A under UNIX to a Tridom Clearlink Plus VSAT Earth Station incorporating a Packet Assembler/Disassembler (PAD), up to an earth satellite, down to the Tridom hub earth station (another PAD), and back to another port on the same Pyramid. Total distance: about 74,000 km (46,000 miles). The delay is 186,000 miles per second (the speed of light) divided by 46,000 miles, or 0.247 seconds, plus the time required for network packet assembly, forwarding, and disassembly using OSI IP and TP4 and Tridom Link Control Protocol over a 512 Kbps synchronous satellite channel, plus any time required for possible error recovery between the PADs (in this case, none). The results agree reasonably well with

Table 11 *Satellite Benchmark, 9600 bps*

Packet Length	Window Size 1	2	4	8	16	31
91	.08	.16	.32	.59	.79	.82
200	.17	.33	.63	.82	.82	
500	.34	.63	.85			
1000	.47	.85				
1500	.55	.89				
2000	.63					

formulas (7), (8), and (9), and they dramatically illustrate the benefits of a combination of sliding windows and long packets in the presence of delays even as short as one second: an elevenfold increase in throughput over basic Kermit!

Trial and error is required to find the best combination of packet length and window size for a particular connection between two computers. Some connections cannot withstand a long barrage of characters — the computers or the communication devices between them may have small input buffers or slow processing speeds that could make them miss packets or lock up. Delay and noise characteristics must be considered too.

How can Kermit's efficiency be improved further?

1. Fine-tune inner program loops, and (in non-windowing versions) prepare the next packet for transmission while waiting for the ACK to the current one.
2. Increase the total packet buffer space to allow larger windows of longer packets, to achieve maximum efficiencies on connections with long delays.
3. Use a *locking shift* for control characters and (in the 7-bit communication environment) 8-bit data to reduce encoding overhead. A protocol extension for locking shifts is in the design stages.
4. Once our computers are fast enough to ensure that processing speed is not the bottleneck, we can devote more CPU time to compression schemes that can increase efficiency not just by 10 or 20 percent, but by *hundreds* of percents.

Future refinements to the protocol and releases of the software will focus on these issues, as well as the related ''human factors'' —automatic detection of the round trip delay, and automatic setting of window sizes, packet lengths, and timeouts according to the delay and noise chracteristics of the connection.

International Character Sets

Christine M. Gianone

Kermit has always been able to transfer text files between unlike computers, even if they have different text character sets, such as ASCII and EBCDIC. To do the text file code conversion, the Kermit protocol requires that local codes be translated into ASCII if necessary before transmission. But ASCII only includes enough letters and symbols for English and a few other languages like German (without umlauts) or Dutch. Kermit can also transfer files containing accented vowels, c-cedillas, etc, as well as files written in Greek, Arabic, Hebrew, Russian, Japanese, or Chinese, but often the special characters will turn into garbage. This is because different vendors use different codes for the same special characters. For example, an Apple Macintosh, an IBM PC, and a DEC VAX use totally different codes for, say, A-diaeresis (Ä), C-Cedilla (Ç), and E-grave (È).

A new 2-level extension to the Kermit protocol permits file transfer to occur in any of a number of standard computer alphabets in addition to ASCII: ISO Latin Alphabet 1, ISO Latin/Cyrillic, ISO Latin/Hebrew, Japanese JIS X 0208, etc. Kermit programs can convert between the computer's local codes and one of these standard alphabets. Level 2 of the protocol extension allows for transfer of files containing a mixture of alphabets, using a set of well-defined alphabet designators and shifts in the data stream.

Over many months, the proposed design was discussed in an international electronic forum, the ''ISO Kermit Group'', which included representives from national and international standards committees and Kermit experts from Belgium, England, France, Iceland, Ireland, Israel, Japan, The Netherlands, Norway, Scotland, Sweden, Switzerland, the USA, the USSR, and West Germany.

Implementation of the Level-1 protocol is a straightforward but laborious process: translation tables are added between each file character set (such as the IBM PC's code pages) and each standard transfer character set (such as ASCII or ISO Latin Alphabet *x*), in *both* directions, then the corresponding SET commands are added along with the protocol that lets one Kermit convey to another which transfer character set is being used. This has been done in MS-DOS Kermit 3.0, C-Kermit 5A, and IBM Mainframe Kermit 4.2, all announced in this issue.

C-Kermit 5A and MS-DOS Kermit 3.0 support ASCII and Latin Alphabet 1 as transfer character sets, and so the text languages that can be transferred with these programs are limited to Dutch, English, Færoese, Finnish, Flemish, French, Gaelic, German, Icelandic, Italian, Latin, Norwegian, Portuguese, Spanish, Swedish, and Welsh. Future releases of MS-DOS and C Kermits will support a wider selection of languages.

IBM mainframe Kermit 4.2 supports several transfer character sets in addition to ASCII and Latin 1, and so its list includes the foregoing languages *plus* Bulgarian, Byelorussian, Greek, Hebrew, Japanese Katakana, Macedonian, Russian, Serbian, Ukrainian, and Yiddish. John Chandler, the IBM Mainframe Kermit author, will gladly add support for other languages not yet supported, such as Polish, Czech, and Hungarian, if someone who has knowledge of the local national variations of EBCDIC will provide him with the appropriate code tables.

John Klensin, chair of the Standards Committee of the Association for Computing Machinery (ACM), says of this work:

> ''The effort to extend the Kermit protocol to deal with multiple international character sets represents one of the first major steps in using existing and proposed International Standards not only to identify what codings have been used, but to permit active conversion among them. Not only is this effort likely to lead to better and more accurate transfer of text and data originating in different countries —an activity that is critical for international data interchange —but it has also identified weaknesses in the relationships among the various Standards and proposals that should be addressed by the Standards-producing bodies.''

Articles from Belgium (p.17) and the USSR (p.18) discuss Kermit's new international character support as it is being used in two very different settings. The current draft proposal (Draft 5) is available on Tape E. It is updated from time to time to reflect reality. In particular, the Level-2 extension may be entirely replaced in light of proposed multibyte universal computer codes like ISO-10646 and Unicode.

If you anticipate adding international character support to a Kermit program, be sure to contact Kermit Development and Distribution at Columbia to be sure you are working from current information.

Report from Western Europe

André Pirard
SEGI, Université de Liège, Belgium

Those who must use international characters in their daily work know how useful the modern computers are that offer them, but those with needs for data communication also know how difficult it is to transfer these characters from one computer brand to another. When these computers began to appear, there was no standard to tell manufacturers which characters to use nor which ''hexadecimal values'' (code points) to assign them to. And so each did it their own way. The result is a Babyl of incompatible character sets and a ménagerie of ad-hoc products, each running on a specific computer to transfer data from, and maybe to, another specific computer. Not a blessing in a mixed environment and always a surprise to the my-computer confident.

The advent of a new standard, ISO 8859, circa 1987, allows these computers to communicate in a ''common language''. The DEC 8-bit multinational character set (MCS) was the starting point for the layout of ISO 8859-1 (''-1'' means it applies to a group of languages known as Latin-1) and, after adoption of an early draft of the ISO 8859-1 standard by at least Microsoft and Lotus, IBM followed with its PC Code Page 850 and mainframe Country Extended Code Pages (CECPs) containing the ISO character set (but at different code points), and Apple said ''I am sure we will keep pace with emerging (and existing) connectivity solutions'' (quoted from private mail).

Kermit is certainly a uniform tool for data transfer in mixed environments. The Kermit designers' response to these problems was very generous and covered international needs widely: I would recommend Christine Gianone's *A Kermit Protocol Extension for International Character Sets* as an invaluable source of information on the theory of standards used to solve the problem, even for Japan. In the simplest case of computers using a single 8-bit character set within each file, the key rule is that they should translate their own character code so that they use ISO 8859 on communication lines and, in general, behave with respect to the outside world as if they were using exactly that code.

John Chandler was quick to implement the EBCDIC CECP to ISO 8859-1 translation (and others!) in his excellent IBM Mainframe Kermit-370, and Joe Doupnik rose to the challenge and added the corresponding features to the already impressive palette of his remarkable MS-DOS Kermit. Given that DEC's MCS is very close to ISO 8859-1, we already have a triangular application of the theory to file transfer. For terminal mode, Joe's Kermit emulates the international version of the DEC VT-320 terminal and can use ISO 8859-1 as well as many ''national'' terminal character sets. And the new Macintosh Kermit being developed by Paul Placeway is far along the same path. When Mac Kermit is able do international text file transfer too, four computer types will raise the possibilities of connections from 3 to 6 (yes, this fast growth shows exactly how ''meshy'' the problem is, and if I count different codes on IBM computers, a conservative number is 100).

People −French, German, Spanish, or just terrestrial *− will use these features.* I can testify to the hard work the Kermit people have done for *you.* Take the few steps to have translation in effect and be convinced the bad old days are over. I am using the new Kermits to transfer French text just as easily now as it has always been for English. And don't hesitate to send them your remarks, félicitations, schönen Dank, or hartelÿk bedankt of any kind. Voilà pour moi.

Although *Kermit News* is not exactly the place, I take the occasion to recommend that manufacturers have their terminals and computer systems use ISO 8859, because null translation everywhere really makes everybody's life much easier −if the programmer knows that a character is coded in eight bits. At least, until there is a single worldwide multibyte code. But that is another story.

[Editorial Note − An international Macintosh Kermit is in development. We endorse André's recommendation and add to it our further recommendation that manufacturers begin to tag each plain-text file with an identifier for the character code in which it is encoded. This will make the job of Kermit and all other applications much easier!]

Mission to Moscow

Kristina Salvatorovna Gianonova

The First International Kermit Conference was held in Moscow, USSR, May 29-31, 1989, sponsored by the International Centre for Scientific and Technical Information (ICSTI). It was attended by over 70 computer specialists from Bulgaria, Cuba, Czechoslovakia, Hungary, East Germany, Mongolia, Poland, and parts of the USSR from ranging from Novosibirsk in central Russia to Tallinn in Estonia.

The conference began with lectures from Christine Gianone and Frank da Cruz on Kermit history, philosophy, use, programming, protocol, and performance, rendered into Russian by interpreters. All attendees were familiar with Kermit, some at the user level, others at the implementation and theory level. On the final day, Gianone's new extension to the Kermit protocol for transfer of text files in diverse character sets (including Cyrillic) was presented; this was of great interest to the international audience, and was warmly endorsed in the ensuing discussion.

The conference concluded with presentations from other attendees, discussing adaption of Kermit programs to Soviet and other East European computers and some of the uses to which they had put these programs. For example, one talk described how Kermit was used in Soviet secondary school CAI labs – 250,000 special ''Kermit PCs'' are being manufactured and delivered to Soviet schools as part of the current five-year plan, each of them with Kermit in ROM. The roster of speakers included: A. Gruzdev, NPO Informatika, Ivanovo; A. Smirnov and V. Rejma, EstNIINTI, Tallinn, Estonia; A. Liberov, Computer Research Institute, USSR Ministry of Radio Industry; M. Motl and F. Zednik, Sigma Concern, Olomouc, Czechoslovakia; Ivo Shmeikal, High Economic School, Prague, Czechoslovakia; G.I. Cherkes, Computer Research Institute, Minsk; Vladimir D. Novikov, VNIIPAS Institute of Automated Systems, Moscow; and Konstantin Vinogradov and Juri Gornostaev, ICSTI, Moscow.

ICSTI was presented with a complete, up-to-date set of Kermit programs and documentation on magnetic tape and diskette. ICSTI in Moscow joins other regional Kermit Distribution centers in England, France, The Netherlands, Japan, Australia, and elsewhere. ICSTI's Kermit Distribution serves the Soviet Union and ICSTI member states, and ICSTI in now a center for coordinating Kermit program development in those countries.

Gianone and da Cruz also visited several other computing installations in Moscow, including VNIIPAS (the Institute of Automated Systems of the National Centre for Automated Data Exchange) and INION (the Institute of Scientific Information for Social Science), and were shown how Kermit plays a central role at each. VNIIPAS has developed an integrated software system including E-mail, conferencing, and data search and retrieval, with Kermit as its communication mechanism.

The First International Kermit Conference was organized and sponsored by Professor A. Butrimenko, Director of ICSTI, Dr. Juri Gornostaev, Head of ICSTI's Computer Department, and the ICSTI ''Kermit Gang'' – Mikhail Morozov, Marina Tumanova, Konstantin Vinogradov, Andrej Yuzhakov, Shamil, and many others.

Kermit Goes to Eastern Europe

Juri Gornostaev and Konstantin Vinogradov
International Centre for Scientic and Technical Information, Moscow, USSR

We are from the International Centre for Scientific and Technical Information (ICSTI) having its headquarters in Moscow. ICSTI pools together efforts of about 300 information specialists from COMECON countries and North Korea in developing national information industries and mutual information cooperation. The authors of this article are professionals in telecommunication systems, office automation, database management systems (and English, see below :-). Once upon a time back in 1984, encouraged by intriguing articles by Frank da Cruz and Bill Catchings in BYTE magazine, we wondered whether we could accustom this overseas stranger, Kermit, to our locale.

History

We received our first tape with Kermit, a DECUS tape, in 1984. Since then our Kermit infatuation started and has never begun to fade or lose supporters. In those days the multitude of computers that this ''frog'' was leaping about did not include the ''Unified Series'' type, widely used in East European countries (Russian abbreviation EC, often referred to as RJAD-computers). In 1985 we received a distribution Kermit tape from Columbia University and decided to adapt it to our mainframe computer EC-1055M (GDR origin). The task was to tune the program to specific telecommunication equipment installed for this computer. Initially, we concentrated on the MVS/TSO Kermit version 1.0 and the first Kermit program produced for an EC computer was based on this software. Shortly, modification of Kermit programs for CP/M, MS-DOS and RSX-11M was carried out enabling their interaction with the newborn Kermit/EC via the exotic equipment.

ICSTI offers on-line database services to many organizations in its Member and non-Member states. So our next ambition was to implement the Kermit protocol for transfer of data search results to our customers. This raised the issue of implementing Kermit programs in the CICS environment. The first version of such a program was developed in 1987 by ICSTI in conjunction with specialists from the Computer Institute (Russian abbreviation IEVT) of the Latvian Academy of Sciences in Riga. IEVT was the leading designer of the ACADEMNET and its specialists, represented by Eugine Mikelevich and Eduard Zinoviev, were great CICS experts. We are thankful for their efforts in initiating the very first version of Kermit/CICS.

Unfortunately this version, by providing access to data sets normally not available through CICS, got too deep into CICS's insides, breaking up its system compatibility. This had its pro's and con's. The user could operate with the ''files'' notion, but there was no possibilty of developing the program and making it portable between OS and CICS versions.

In 1988 we attempted to write a Kermit/CICS version that would use only the standard system conventions of CICS/VS to access data sets. The younger ICSTI generation, namely Alexander Resaev, was successful in the undertaking. This particular version was presented to Christine Gianone and Frank da Cruz during their stay in Moscow at the First International Kermit Conference held in ICSTI in May-June 1989. The version takes into account particulars of our local equipment. Therefore, slight changes could be necessary in order to use this program on ''standard'' equipment.

Along with managing the mainframe problem, ICSTI specialists developed Kermit programs for USSR and GDR-origin micro-computers. At present Kermit runs on such computers as Robotron 1715, Robotron 1715M (CP/M compatible), Iskra 1030, EC1840, and EC1841 (MS-DOS compatible). As all of these programs were based on old versions and are of interest mainly to Soviet users, ICSTI distributes them on a separate magnetic tape (as well as diskettes).

Kermit and Cyrillic Character Sets

At least three Cyrillic character sets may be installed and used on PCs at the present time. First is the ISO 8859-5 international standard Latin/Cyrillic set. The Iskra 1030 and EC1840 (CGAs) manufactured in the USSR are supplied with built-in ISO Cyrillic. EC1840 and EC1841 (EGAs) are capable of loading any character set also. With this option available most of our programmers prefer to use ''Alternative Cyrillic'' which contains Cyrillic letters in columns 8, 9, A and E of the code table, very similar to IBM PC Code Pages. This Cyrillic character set was proposed by the Computer Centre of USSR Academy of Science in 1987 and since then has been widely used in the USSR. Moreover, most of the Western PCs supplied to the Soviet Union have Alternative Cyrillic incorporated and/or invoked by means of different kinds of drivers. And one more Cyrillic set known as ''Bulgarian'' may be used on PCs manufactured in Bulgaria or by the software developed by our Bulgarian colleagues. In this case columns 8, 9, A and B contain Cyrillic letters.

In practice, neither of these Cyrillics are widely used ''on-the-line'' today. In terms of Kermit these are FILE CHARACTER-SETS. The most popular Cyrillic coding used in data communication is the so called KOI-7 character set, which is ''compatible'' with itself only. This set requires the Shift-In/Shift-Out technique to switch between Latin and Cyrillic graphics (GL and GR respectively). Usually, KOI-7 is generated by our mainframe telecommunication system. But historically, to avoid SI/SO in some applications an ''exotic'' character-set was invented by programmers. Cyrillic letters were coded in lowercase Latin ''by sound''. Furthermore, many terminals were designed to display lowercase Latin as uppercase Cyrillic! As a result, this character set consists of uppercase Latin and uppercase Cyrillic letters. It was never registered anywhere but is still in use, as well as (and more than!) KOI-7, and specialists often refer to it as ''short KOI''.

After all these words you may understand our enthusiasm receiving the Draft *[Ed. – See p.16]*. During our early work with Kermit programs we also made attempts to put this Cyrillic chaos in order and establish a convention for transfer of Cyrillic files. We succeeded in this and moreover the programs work !-). Our convention for Cyrillic transfer in Kermit programs is valid for both terminal emulation and file transfer. We wanted to achieve results quickly and at low cost. That is why the convention was based on use of KOI-7 and ''short KOI.'' Now, two years later, we clearly see the disadvantages of such an approach.

The fundamental approach proposed by Christine Gianone resolves the problem which we have in current versions of our programs and this makes it a notch higher. That is why we are deeply interested in the success of this ''international'' work.

Unfortunately, the last draft of the Kermit extension proposal, Draft 3, skips commands for terminal emulation, but strange though it may seem this particular question is nicely resolved in the very first implementation of this Kermit Protocol Extension by MS-DOS Kermit v3.0. This program allows any kind of Cyrillics on the line and converts it to the character set used by the PC. Moreover, it's possible to mix incoming KOI-7 and ''short KOI'' in case you are not sure which kind of them the remote host prefers. But Cyrillic file transfer is still a problem. Transmission of such files is possible in ''transparent'' mode only when it's possible to turn on ''the same'' Cyrillic for both Kermit programs.

[Editorial Note — This is only because the necessary translation tables between ISO Latin-5 and the Cyrillic file character sets have not yet been added to MS-DOS Kermit!]

Kermit Applications at ICSTI

The wide range of Kermit applications at ICSTI first of all includes downloading of documents from mainframe databases to PCs. As a rule, the volume of data transfer in this mode is 1-1.5Mb per file. The documents received via Kermit are loaded into the PC retrieval system and are supplied on diskettes to our subscribers.

ICSTI has a ring LAN based on RS-232C. Here Kermit is used for file exchange between different kinds of computers connected to this LAN.

Kermit is also widely used for connecting to remote hosts in terminal emulation mode. For example, MS-DOS Kermit provided access to Western on-line services during the work of the First East-West Online Information Meeting, jointly held at ICSTI with ''Learned Information'' in October 1989 and proved to be a success. Particularly, the excellent operation of the script language enabled us to simplify to a great extent the process of connection via ICSTI LAN to X.25 PAD and further via X.25 IASNET network to remote hosts and services (IASNET is a network maintained and developed by the Institute of Applied Systems of the USSR Academy of Sciences in Moscow). Connection and message transfer in E-mail services of EARN/BITNET and IASNET networks is also carried out with Kermit. Message acceptance and forwarding is done with Kermit's SEND/RECEIVE or TRANSMIT commands.

And still one more Kermit application was introduced at ICSTI in January 1990: a Kermit Server has been installed in our X.25 network for distributing the Kermit program versions most popular in the USSR. Anybody can receive the Kermit updates offered by ICSTI through IASNET network address 02502-120702. Since all ICSTI member states are connected to IASNET, they can use our host to obtain the latest materials on ''Kermit Culture''.

ICSTI Kermit Distribution

ICSTI distributes Kermit software on six magnetic tapes: the A-E tapes are copies of the original Columbia University distribution tapes, and the F tape contains all Kermit adaptations carried out at ICSTI or other organizations in the USSR. Kermit software for microcomputers is distributed on diskettes also. To request Kermit materials, the user fills in an order form resembling that of CUCCA, stating the required materials, including copies of documentation in Russian and English. So far around 100 organizations have received Kermit distribution materials from ICSTI.

Kermit Promotion

At present the Kermit enthusiasts at ICSTI (The Kermit Gang) are working on the Russian translation of the popular Kermit book written by Frank da Cruz, to be published by the Soviet NAUKA publishing house. Our next step will be the book *Using MS-DOS Kermit* by Christine Gianone, recently in print and which we are sure will be popular as well.

Kermit Conferences

The First International Kermit conference, held at ICSTI in Moscow, May-June 1989, brought together more than seventy information specialists from ten countries (p.18). This event was a great success and we dream of a Second International Kermit conference. We suggest organizing this forum in 1991 with wide participation of countries both from the West and the East. We are open for proposals on the timing and location (Moscow, Suzdal, Vladimir, or the North Pole). Please don't hesitate to send us your thoughts!

Conclusion

Finally, we would like to express our sincere wish and readiness to diversify our direct contacts with all the members of the Kermit family. Our postal address is ICSTI, 21-b Kuusinen Street, 125252 Moscow, USSR. We can be also reached by E-mail through EARN/BITNET address `ENIR@IAEA1`.

Translation by Marina Tumanova,
K.G.M. (Kermit Gang Member)

Kermit's in the (US) Navy Now

Frank Dreano, Jr.
Navy Management Support Office
Chesapeake, Virginia, USA

I am a government employee working on the Navy's NALCOMIS application (Naval Logistics Command Management Information System). This is a huge application running on a network of Honeywell DPS-6 super-minicomputers. One of the functions of NALCOMIS is to issue materials to repair all types of Navy aircraft (F-14, A-6, etc.). NALCOMIS can request the issuance of material, but the requisition must go to one of the Navy's existing supply systems for financial accountability purposes and a status must be returned by these systems. This has to occur before NALCOMIS can ''really'' issue the part.

Unfortunately, these other supply applications operate on IBM, Tandem, Burroughs, VAX, and other Honeywell systems at distances of up to 300 miles from NALCOMIS sites. Therefore, until recently, a batch magnetic tape database synchronization process was being used to interface NALCOMIS to remote Navy supply systems. Because this process caused application down-time, these tape exchanges only occured at intervals of up to 72 hours (provided the tapes were good!). This lag time between an item being requisitioned and a remote status being returned directly affected the operational readiness of Naval fighter squadrons.

The job of providing a better interface between NALCOMIS and these various systems was given to me. I spent a year investigating and testing different telecommunications protocols such as host-to-host 2780, 3270, SNA, HASP synchronous and various asynchronous protocols, not to mention looking into a Wide Area Network (WAN) solution. These methods of telecommunication all seemed inappropriate for different reasons. For instance, (1) not all the systems involved ''agreed'' on what constituted a specific protocol, (2) many of these options would require expensive dedicated and conditioned leased-lines and were really ''one-way'' remote job entry style protocols, (3) an integrated system test between my activity (in Norfolk, Virginia) and the activity that developed the other applications (in Mechanicsburg, Pennsylvania) would be a logistical and bureaucratic nightmare.

During this time, in attempt to learn the C language, I wrote a Kermit program for the Honeywell minicomputer. Once I had this program operational, I was struck by its ease of use and versatility. (It also turned out to be 3 to 4 times faster than the proprietary mainframe-to-micro file transfer programs we had been using!) I then decided, in the face of some heated controversy, that Kermit was going to be the NALCOMIS generic means for electronically interfacing to *all* supply applications regardless of the hardware on which they operated. Within two weeks of this recommendation being approved I completed an integrated systems test of a prototype kernel for an unattended Kermit telecommunications interface between Honeywell and Tandem computers. The rest, as they say, is history.

The various electronic interfaces using Kermit have now *eliminated* the 72-hour batch processes described above. Status returns to NALCOMIS from remote supply systems now occur in 7-10 seconds. This method of one-size-fits-all telecommunications to various systems has the following significant advantages over ''host-to-host'' or totally vendor-specific protocols:

Cost: The majority of the interface development, prototyping and implementation costs to the U.S. Navy consist of basically the price of an 80286-based PC and my time. Since NALCOMIS will operate at over 200 sites world-wide the savings is significant.

Development: (1) The activity in Mechanicsburgh develops whatever telecommunciations file transfer protocol they like between their supply system and an MS-DOS PC (usually Kermit!). I, in turn, have already developed the Honeywell Kermit and PC `.BAT` file at my site in Norfolk. We eventually all meet once at the first prototype site for each style interface and mesh our respective programs. (2) An additional unexpected advantage was the ability to ''massage'' the data that is being passed back and forth on the PC. Several ''show-stopper'' application interface bugs were fixed this way on-site. Moreover, since the old batch interface application software dealt with files produced from magnetic tape, the slight application changes required to use files produced by Kermit were a breeze.

Life cycle support: Basically, Columbia University is providing 90% of the life cycle support. Every time Kermit is improved, these interfaces improve. I recently implemented 2000-byte packets into the interface environments in about two days.

Throughput: This was initally my largest concern and it turned out to be totally unfounded. At 9600 bps (with the potential to run at 19,200 bps), with data compression, the interfaces can deliver data faster than the respective applications can consume, massage it and disgorge responses. In addition, since the interface is generally driven by interactive users requesting parts, transmissions tend to be short and bursty. This eliminates a lot of the concerns of large, bulk file transfers. Even so, the first site (now operational for six months) transfers about a megabyte per day of interface data. NOTE: All interface data is journaled to an *extra* hard disk on the microcomputer to provide automatic restart/recovery in the event of a PC head-crash or a mainframe glitch.

Operational aspects: The Kermit/micro philosophy of performing these interfaces had several advantages over other methods. For instance, if either NALCOMIS or the supply system is down, the PC can buffer interface data until the other system can come back online. Also, the interface does *not* require the application to be taken down as the old batch system did. Honeywell pipe files (the equivalent of ''named FIFOs'' in UNIX parlance) were used to replace the old files produced by magnetic tapes and provide asynchronous inter-process communication between Honeywell Kermit and the NALCOMIS application. Additionally, impact on both the supply and NALCOMIS applications is about equivalent to that of another interactive user. The interface PC basically ''logs in'' to both mainframe systems upon boot-up, invokes the respective Kermit servers (or other communications software) and starts the interface get/send loop until operator intervention or a catastrophic hardware failure.

Site computer operators love this interface. Instead of dealing with magnetic tapes from foreign systems and hearing users complain about system down-time, they have an easy-to-understand, micro-based electronic interface. As an amusing aside, the first prototype site that was implemented agreed to let us ''stand-up'' the electronic interface for five working days as a favor. After that time we were to go away and let them resume their batch interfaces while we fixed whatever program bugs, hardware problems, etc., that we had discovered. At the end of the five days the site literally would not allow the electronic interface to be removed and has operated with it day and night for the past half year.

The interface micro has a graphics window package (also public domain) that shows what the interface is doing (direction of data transfers, number of round trips completed, etc.) at any instant in time. Moreover, interface activity and accuracy reports have been moved from the already heavily-loaded mainframe systems to the interface micro for execution.

In summary, I simply cannot say enough good things about Kermit. It has provided the perfect solution for an automated, low-to-medium volume, flexible telecommunications requirement, when other more exotic methods would probably have been expensive overkill. Kermit has made the harried life of a systems programmer (I am really *not* a telecommunications guru) much easier. What a pleasure it is when someone says to me ''yeah, but can NALCOMIS interface with a Belchfire-8 system . . . '' to respond by saying ''When would you like the software delivered?''

FAST FOOD!

Roman M. Lubynsky, Nashua, New Hampshire, USA

Fasfax Corporation is a mid-sized manufacturing firm based in New Hampshire with about 250 employees. Fasfax sells and services microprocessor based point-of-sale (POS) systems for the quick service restaurant industry. Fasfax has more than ten thousand installations. Major Fasfax users include Burger King, Kentucky Fried Chicken, White Castle, Carl Jr.'s, Whataburger, and Skipper's Seafoods.

POS systems perform quite a number of functions within a restaurant. All customer orders are entered into the system through one of the terminals (a cash register). As customer orders are entered, the POS system displays elements of the order on remote terminals and printers at various production stations where the food is prepared. In addition to keeping track of sales dollars, complete product mix information is recorded. Employees also use the system as a time clock. Management reports are available that integrate all of the above information to provide comprehensive and detailed productivity and performance information. The POS system today is a crucial and vital part of almost every fast-food restaurant. Fasfax POS systems are based upon a proprietary hardware and software architecture. Both the POS operating system and applications language were developed by Fasfax. The language is called ''STRING''. It has many similarities to FORTH and runs interpretively. The operating system provides real-time transaction processing in a multiuser, multi-tasking environment.

Large Fasfax users have used data telecommunications to transfer restaurant operational information from individual store POS systems to their home office mainframe host systems since the mid 1970s. In 1983, Fasfax implemented a Kermit server function within their POS software in addition to the other protocols supported. This was done to provide a communications protocol that had many implementations running on a large number of hardware and operating system platforms.

The Fasfax version of Kermit runs in server mode and may be dialed at any time. Password security is provided. Once connected to the POS system, the host may request various types of operational data to be sent from the remote POS. Information may be requested on a real-time basis displaying current restaurant activity during operating hours, or data files containing summary data may be collected after the end of the business day. Polled data becomes the input into the management information system used by headquarters.

Electronic mail can be sent from HQ to the store to print out messages for the restaurant manager. Information in the POS data base may be modified or replaced by downloading new prices, new menu item information, or even a new POS program version. Fasfax also can use Kermit to perform remote diagnostics and software maintenance for its users.

Kermit has proven to be extremely popular as more than 2,000 individual restaurants are polled daily using Kermit from their respective headquarters. A wide variety of host systems are used —VAX, PDP-11, HP, IBM, and many MS-DOS based PC's used by the smaller franchisees with 50 or fewer units. Kermit provides a high degree of reliability with overall success rates of over 99% —even in locations like Hawaii where a franchisee has about 30 Burger Kings spread over many islands and communications are often noisy. Kermit is utilized in many remote locations in the U.S. where new TELCO switching systems are yet to be installed. There are also a number of systems in the Caribbean and in Europe where line interference and long-delay satellite communications are no problem for Kermit to handle.

So next time you get to the head of the line at your favorite fast-food restaurant, remember there could be a little Kermit in the cash register helping to make sure that your order comes out right.

Kermit Protocol in Manufacturing

Golden E. Herrin, Control Product Specialist
Cincinnati Milacron, Cincinnati, Ohio, USA

As computer-integrated manufacturing expands, users and systems integrators alike continually search for lower cost reliable methods of getting computer equipment to talk to each other. Kermit File Transfer Protocol is starting to appear on the manufacturing floor, filling the gap between low cost serial connections without error checking and high cost, full capability local area networks (LANs).

There are many interface solutions available, including Manufacturing Automation Protocol (MAP). The cost of utilizing these various interface methods is generally proportional to their capability and speed with LANs at the top end of the price range. The lowest cost solution to connecting computer equipment is still a serial connection provided by RS-232, RS-422 or RS-423 running very basic protocols with error checking. Most CNCs and programmable controllers today include one or more of these interfaces as standard equipment. Kermit File Transfer Protocol implemented on these controls upgrades the existing interfaces and data lines to a packet transmission type system, with checksums and retransmission capability to enhance data integrity.

The proliferation of Kermit into the shop is primarily due to the wide variety of computers on which the protocol has been implemented. These computers when used as direct-numerical control (DNC) hosts, cell controllers, front end processors, programming systems and status gathering systems have made available a low cost interface protocol for connecting into shop floor equipment. Kermit is desirable since it runs on the existing serial data lines. Another plus is emerging out of the binary Cutter Location (BCL) environment. Because of its packet transmission capability Kermit has become a very desirable protocol for transmitting BCL data to BCL input controls.

Even though Kermit is implemented on a wide range of computers, it is still not available on all shop floor control devices, but there is a sizeable list. The following control devices offer a Kermit protocol interface: Cincinnati Milacron's Acramatic 750 and 950 CNC Controls; GEFANUC's Series 0, 10, 11, 12 and 15 CNC offer Kermit in the MMC development mode; Siemen's RCM-3 Robot Controller; GRECO System's Versifile, Versinet and Greco-Net Products; BCL Technology's BCL Front End; and the ISS Vega Series 9000 CNC.

The role of Kermit in manufacturing is not clear yet. It could be that Kermit will serve as a short term solution until MAP products become available or, it could become a long term low cost solution coexisting with MAP. Much depends on where MAP interface costs settle out.

Excerpt printed with permission from CIM Perspectives, Modern Machine Shop, November 1989

MS-DOS Kermit 3.0 and Radio Communications

Barbara Cox, Proxim Inc.
Mountain View, California, USA

Proxim manufactures spread spectrum radios and data communications networks for commercial and industrial applications. Proxim's focus is on providing a low cost and very reliable wireless high data rate communications link that does not require FCC site licensing. Spread spectrum is a form of radio transmission that ''spreads'' a signal's spectrum over a radio frequency channel greater than that necessary to transmit the information.

These radios provide numerous advantages. The radios have a very small form factor, which make them ideally suited for use in products that are portable, or where space is otherwise constrained. The radio's low power consumption makes it suited for applications with limited input power available. The radios are multichannel and have a data rate of 122Kbps for applications with high data rate requirements. The spread spectrum radio product family can operate at this data rate at ranges up to a quarter mile.

With the new radios in mind, our goal was to come up with an effective way to support sales demonstrations of our wireless products. We decided to show wireless data communications between two personal computers running Kermit over the radios.

MS-DOS Kermit 3.0 was selected because it is able to support the radio's half duplex mode of operation. Since MS-DOS Kermit 3.0 can use RTS/CTS handshaking for half-duplex line turnaround, we were able to control the radio's receive and transmit modes via the interface cable between the radio and the computer. When RTS was asserted the radios would transmit data, and when RTS was deasserted the radios would receive data.

Another goal of the sales demonstration was to have the radios work with standard, off-the-shelf computers and software (Kermit). A Mitsubishi PC/AT desktop clone and a Compaq LTE/286 laptop computer were configured with the radios and Kermit. The computers communicated asynchronously through the RS-232 serial port to the radio's synchronous port. While the Mitsubishi computer was in server mode at a stationary position, the mobile Compaq was issuing remote commands to the server at various distances, demonstrating portable wireless communication.

This not only exercised the radios, but Kermit as well. Testing at various distances and environments helped determine the operating parameters, and how well Kermit and the radios performed. With the server computer in one office of the building, the Compaq continually sent and received data from different offices, floors, and outside the building. Communication was successful up to 38,400 bps on the computers in asyncronous half duplex mode, at packet sizes varying from 50 to 2000 bytes. The radios can support a data rate of 122Kbps in synchronous mode.

The development of the sales demonstration package revealed a number of application possibilities in which spread spectrum radio communications can be very beneficial. The radios lend themselves well to markets wanting low cost wireless communication with high data rates, portability, low power consumption, and no FCC site licensing. Local area networks, the portable data recorder (PDR), and medical industries are just a few examples of potential spread spectrum radio communication applications.

Radios in local area networks provide connectivity without the wires. Replacing wired networks with spread spectrum radios not only eliminates wiring, but also can provide higher data rates with additional security, reliability and error checking.

The use of radios in PDRs offers tremendous benefit in meeting the needs of warehousing, real-time inventory, production control, and quality control. Also included is maintenance and many other mobile and remote data collection and communication applications.

In an effort to contain health care costs and compensate for reduced nursing staff, radios can be implemented in a wireless data network. Vital statistics can be collected and communicated via radio from medical equipment to nurses' stations, where patients can be centrally monitored. Economic gains can be quickly realized for systems looking to improve performance versus hardwire retrofitting.

Kermit has been a tremendous asset in demonstrating and testing our radio's capabilities. As further testing and experimenting progresses, Kermit may prove itself viable for many commercial and industrial spread spectrum radio communication applications.

Kermit Is Not a Toy!

Landy Manderson, Lead Software Specialist
University of Alabama at Birmingham,
Birmingham, Alabama, USA

We really appreciate the work that the folks at Columbia do towards keeping Kermit alive and well! Maintaining the Kermit library and the Digest and the protocol is a big job, but it definitely ensures that this ''volunteer'' effort keeps on track.

Now, as for the people who think Kermit is just a toy used by hobbyists . . . HA! These people have obviously not visited *this* campus. The University Computer Center (my department) maintains all the primary administrative records for the University (including the payroll, purchasing and accounts payable, student information, and more) on an IBM mainframe. TUCC only ''supports'' two mechanisms for accessing all this information: 3270 coax and Kermit.

Years ago, our mainframe operations group began installing asynchronous statistical multiplexers and protocol converters so that people outside the range of 3270 coax could get to the mainframe. Of course, initially they only used ''dumb'' terminals, but soon the PC's needed to connect also. The operations group started out recommending Crosstalk, but we (the User Services section) went looking for a better option. We chose Kermit because it was (a) free, (b) available on many machines, (c) free, (d) non-commercial, (e) free, (f) able to do file transfer with our mainframe, and (g) free.

Today, when a personal computer user wants to access any of our systems for any reason (administrative, academic, research, or E-mail), we give them a copy of Kermit and an appropriate handout showing how to use it. Of course, we have a few die-hard users that just *have* to use their *own* software, but there is always one catch: they *have* to be able to set it up for Kermit file-transfer, because we *only* have IND$FILE and Kermit file transfer protocols on the mainframe.

For the future, UAB is attempting, like many institutions, to come up with a campuswide networking plan. We have already done some preliminary drafts of RFI's and RFP's for a campus communications hub. One of those drafts states: ''[The hub] must include facilities for access by users who cannot buy *anything* . . . software (such as Kermit and NCSA Telnet) must be usable for base services.'' It is not official yet, but you can see how important we consider Kermit to be in our future plans. Thank you again for your work and support.

MS-DOS Kermit Backers

Joe R. Doupnik

While the development of each issue of MS-DOS Kermit includes contributions of code and ideas from a number of people around the world, several organizations have also contributed in the form of software or hardware. These material assets have allowed me to enhance Kermit for wide groups of users. If it seems peculiar to list an individual and my University please note that they incurred significant real costs to themselves while assisting MS-DOS Kermit development. I offer my appreciation and thanks to each contributor.

Commercial Contributors:

- AT&T Networking Group. Unix PC development machine, STARLAN and STARGROUP networking software, STARLAN boards.
- IBM and Yale University. EBIOS, YTERM, and LANACS async server software.
- Novell. Excelan division: LAN WorkPlace for DOS software and EXOS 205T Ethernet board. Educational division: NetWare file server software NW/VMS, NW/386, ANW/286 SFT II, ELS II, NASI/NACS async server, PC AnyWare, Ascom IV, network boards, and more over the years. Part of the Technology Transfer Partnership, preceeding and succeeding.
- Tseng Laboratories. Ultra Pak Short display adapter board.

Non-commerical contributors:

- Terry Kennedy, St. Peter's College. DEC VT340/330 Technical Reference manuals.
- Columbia University, Kermit Project. Hayes 1200B modem, books, specs.
- Utah State University. Electrical Engineering Department for long distance phone charges. Dean of Engineering for supporting VMS Mail for many years. Computer Services for VMS Mail costs and now a VMS MicroVAX.

And special thanks to the people who contributed so much to the development and testing of MS-DOS Kermit 3.0: Kevin Black, Phil Benchoff, Susan Bramhall, Jack Bryans, John F. Chandler, Frank da Cruz, Drew Derbyshire, Hirofumi Fujii, Christine Gianone, R. Brooks Van Horn, John Junod, Terry Kennedy, Mikko Laanti, Joseph Moyer, Dan Norstedt, André Pirard, Matthias Reichling, Fred Richter, Gisbert W. Selke, Ted Shapin, Gary Stebbins, Bert Tyler, Konstantin Vinogradov, Paul Whitmer, and Mark Zinzow.

The Kermites

Ken Suh and Max Evarts
Kermit Distribution, Columbia University

You may have wondered who the people are behind those voices that you hear when you call Kermit Distribution. There are currently two Kermites, as we are affectionately named, who take care of Kermit software distribution. Ken Suh, who recently graduated from Columbia College with his Bachelors Degree in Political Science, is an experienced Kermite. He has been working at Kermit Distribution since July, 1988. He will be with us until the summer of 1990, after which he plans to enter public service. Max Evarts joined Kermit in December, 1988. He is a happily married man who is currently trying to finish a BA in English Literature as a part-time student at Columbia University. Both work full-time at Kermit Distribution. Those of you who have been dealing with the Distribution Center for a while may realize that the current arrangement, with two full-time employees, is a change from the past. Prior to Max's joining Kermit, the Center was run by one full-time person and several part-time workers. The move to two full-timers was made in order to make the Distribution Center more independent and to relieve Chris of the burden of juggling part-time schedules. We will, however, keep on one part-time student worker, Taka Sajima. Taka has been working part-time for Kermit during the school years since April, 1988. Hopefully, you will also notice the benefits of this arrangement, such as increased efficiency and continuity.

Incidentally, if you call the Kermit Tech Support Line you will talk to one of us as well. Despite our apparently Liberal Artsish educational goals, we are both avid computer fans and handle the first level of technical support for Kermit. Kermit Distribution provides an excellent environment for learning about computers and data communications. On a daily basis, we use some fairly advanced features of UNIX to make our work easier, and we are always looking for ways to let the computers do more work, more efficiently. One cannot help but learn quite a bit about Kermit, and data communications in general, while working here. In time, one comes to recognize the ''Ten Most Commonly Asked Questions,'' and can easily answer them. Working with Chris and Frank is instructive in itself. So feel free to try us out on a question, we will try our best to help you! If we can't, we'll pass your call along to someone who can.

There, now we are no longer mere disembodied voices at the other end of a long-distance phone connection!

Have You Seen . . . ?

Kermit, a File Transfer Protocol, Frank da Cruz and Bill Catchings, BYTE Magazine, June & July 1984.

Kermit, A File Transfer Protocol, Frank da Cruz, Digital Press, 1987.

Kermit Software Capable of Linking Diverse Systems, Don Steinberg, PC Week, March 3, 1987.

Kermit: All-Around Micro File Transfer, Lynn Jarrett, Digital Review, April 6, 1987.

Shopping for Software that Lets PCs Chat with Mainframes, Frank da Cruz and Christine M. Gianone, Data Communications Magazine, Dec 1987.

Terminal Emulation Makes Kermit a File-Transfer Contender, Steve Higgins, PC Week, Aug 8, 1988.

MacKermit Transfers Files among Minis, PCs, Mainframes, Jonathan Perrow, MacWEEK, June 21, 1988.

Making PC Software Work with Digital PBXs, Christine Gianone and Frank da Cruz, Data Communications Magazine, October 1988.

Algoritm Druzhbi (''The Friendship Algorithm''), Andrei Yuzhakov, POISK (Newspaper of the USSR Academy of Sciences), June 1989.

Kermit to Speak non-ASCII: Greek, Russian, et al., Sue J. Lowe, Data Communications Magazine, June 1989.

Kermit in the USSR, EDUCOM Review, Fall 1989.

It's Time to Prepare for International Computing, Christine M. Gianone, PC Week, October 2, 1989.

Using MS-DOS Kermit, Christine M. Gianone, Digital Press, 1990.

The Matrix, John Quarterman, Digital Press, 1990.

Who Is Kermit and Why Is He in My Computer?, Bob Rasmussen, NCR Monthly, March 1990.

More Than a Frog About Town, Kermit Is Serious Software, David Buerger, Info World, March 12, 1990.

Neue Kermit für MS-DOS-Rechner, MC (Die Mikrocomputer-Zeitschrift), April 1990.

Ordering Information

Kermit software is distributed by Columbia University on diskettes, magnetic tape reels and cartridges in several formats. Tapes include source code and documentation in machine-readable form for each Kermit implementation, and in some cases also binaries (encoded in hex or other printable format). Selected Kermit versions in formats that Columbia cannot produce are available from diskette services, user groups, and volunteers.

Kermit programs are collected on five reel-to-reel 9-track tapes: A, B, C, D and E. The programs are assigned to tapes A-D as shown in the second column of the Kermit version list as follows: Popular Micro and PC versions on Tape A (*exception:* C-Kermit versions, including those for the Macintosh and Amiga, are on Tape B), Popular Mini and Mainframe versions on tape B, and less popular micro/PC and mini/mainframe versions on tapes C and D respectively. Tape E contains machine-readable copies of the Kermit User Guide, Protocol Manual, articles, the Info-Kermit Digests, Newsletters, the International Character Set Proposal and other general documents. Tapes and cartridges are available in these formats:

ANSI: ANSI labeled ASCII, format D (variable length records), blocksize 8192, 9-track, 1600 bpi. Readable by many computer systems, including VAX/VMS.

TAR: UNIX TAR format, blocksize 10240, 9-track, 1600 bpi.

OS: IBM OS standard labeled EBCDIC, format VB (variable length records), blocksize 8192, 9-track, 1600 bpi, for MVS, CMS, and other mainframe systems.

TKV: TK50 tape cartridge for the DEC MicroVAX. VAX/VMS BACKUP format. Also readable by TK70. Each TK50 cassette holds three Kermit tape sets: A, B, and E, or C, D, and E.

TKU: TK50/TK70 for the DEC MicroVAX. Ultrix TAR format. Three tape sets per cartridge.

SUN: UNIX TAR-format 1/4-inch tape cartridge. Readable on SUN computers, IBM RT PC, and possibly others. Three tape sets per cartridge.

NO OTHER TAPE FORMATS ARE AVAILABLE. If none of our formats looks familiar to you, then specify ANSI −this is an industry standard format that *should* be readable by any computer system (ANSI tape specifications are provided on paper). IBM VM/CMS users should order the OS format and use the program OSTAPE, which is included, to read the tape on your CMS system; instructions are included with the OS tape.

TERMS AND CONDITIONS

The Kermit software −including source code −is furnished without software fee or license, and without warranty of any kind, and neither Columbia University, nor the individual authors, nor any institution that has contributed Kermit material, acknowledge any liability for any claims arising from the use or inoperability of Kermit software. Since source code is available for all implementations, users may fix bugs, make improvements, or write documentation where it is lacking, and are encouraged to contribute their work back to Columbia for further distribution. Under certain conditions software or hardware producers may include Kermit protocol in or with their products. Contact Kermit Distribution for further information.

Although the Kermit software is free and unlicensed, Columbia University cannot afford to *distribute* it for free. To defray our costs for media, printing, postage, labor, and computing resources, we require moderate distribution fees from those who request Kermit directly from us. The schedule is given on the order form. You may also obtain Kermit software from many other sources, including user groups, networks, dialup bulletin boards, and you may copy them from friends, neighbors, and colleagues. In fact, you may obtain Kermit programs from anyone who will share them with you, just as you may share them yourself. Once you have obtained Kermit software, *please use it only for peaceful and humane purposes*.

TO ORDER KERMIT FROM COLUMBIA UNIVERSITY, fill out the attached order form. PREPAYMENT BY CHECK is encouraged; an additional ORDER PROCESSING FEE is required if we must issue an invoice. Orders are shipped by delivery service or US airmail, normally within 2-4 weeks of receipt, but firm delivery schedules or methods cannot be guaranteed. Prices are in US dollars and include shipping costs. When two prices are shown (like $120 / $155), the first price applies to North America and the second is for shipments outside North America. *Rush service* is available for an extra fee. Call (212) 854-3703 for additional ordering information.

Telephone orders can not be accepted, nor can payment by credit card or bank transfer.

KERMIT VERSIONS Listed by Machine and Operating System (A-H)

Prefix	Tape	Machine	Operating System	Program Language	Program Version	Released yy/mm/dd
CK	B	(various 68000)	OS-9	C	4E 72	89/02/02
OS9	C	(various)	OS-9	C	1.5	85/09/20
FLX	C	(various 6809)	Flex 9	6809 Asm	-	86/04/17
UCP	C	(various Pecan)	UCSD p-Sys	Pascal	1.1	89/07/05
CX	A	(various)	CPM80	LASM	4.09	87/08/06
TRI	C	(various)	TRIPOS	BCPL	-	87/07/10
UX	B	(various)	Unix	C	3.0	84/08/02
CUC	D	(various)	Unix 2xBSD	C	-	86/04/10
CUC	D	(various)	Unix 4xBSD	C	-	86/04/10
CK	B	(various)	Unix Sys 3	C	4E 72	89/02/02
CK	B	(various)	Unix Sys V	C	4E 72	89/02/02
CUC	D	(various)	Unix Sys V	C	-	86/04/10
CP	A	(various)Generic	CPM80 2.2	LASM	4.09	88/05/17
CP	A	(various)Generic	CPM80 3.0	LASM	4.09	88/05/17
MS	A	(various)Generic	MS-DOS	MASM	2.32A	89/01/24
UF	C	6809	UniFLEX	C	1.4	89/02/02
CP	A	Access Matrix	CPM80 2.2	LASM	4.09	88/05/17
BBC	C	Acorn BBC B	OS1.20	6502 Assem.	1.45	87/05/19
BBC	C	Acorn BBC B+	OS 2	6502 Assem.	1.45	87/05/19
BBC	C	Acorn BBC B+128	OS 2	6502 Assem.	1.45	87/05/19
CP	A	Acorn BBC Micro	CPM80 2.2	LASM	4.09	88/05/17
CP	A	Acorn BBC Torch	CPM80 2.2	LASM	4.09	88/05/17
BBC	C	Acorn Compact	OS 3	6502 Assem.	1.45	87/05/19
BBC	C	Acorn Compact	Panos	C	4C 57	87/07/14
BBC	C	Acorn Master 128	OS 3	6502 Assem.	1.45	87/05/19
AC	C	AcornWorkstation	PANOS	C	-	87/07/13
MS	A	ACT Apricot	MS-DOS	MASM	2.30	88/05/12
CP	A	Action Discovery	CPM80 2.2	LASM	4.09	88/05/17
CK	B	Alliant FX/8	Concentrix	C	4E 72	89/02/02
AM	C	Alpha Micro 68K	AMOSL 1.2	AM68K Asm	1.0	85/02/11
CK	B	Altos 986	Xenix 3.0	C	4E 72	89/02/02
CP	A	Amstrad CPC 6128	CPM80 2.2	LASM	4.09	88/05/17
CP	A	Amstrad PCW 8256	CPM80 2.2	LASM	4.09	88/05/17
APX	C	Apollo	Aegis	Pascal	2.9	89/05/07
CK	B	Apollo	Aegis	C	4E 70	88/01/27
CP	A	Apple II	CPM80 2.2	LASM	4.09	88/05/17
APP	A	Apple II	DOS,ProDOS	CROSS	3.86	89/10/31
UCA	C	Apple II	UCSD p-Sys	UCSD Pascal	1.0	86/04/08
CKM	B	Apple Macintosh	Mac OS	MPW C	0.940	88/05/26
CN8	C	Argos Pro PC	CCPM86	ASM86	2.9	86/04/10
CK	B	AT&T 3B Series	Unix Sys V	C	4E 72	89/02/02
MS	A	AT&T 6300	MS-DOS	MASM	3.0	09/01/16
CK	B	AT&T 7300 UnixPC	Unix Sys V	C	4E 72	89/02/02
ATA	A	Atari Home Comp.	DOS	Action!	-	84/01/09
UCP	C	Atari MEGA ST2	UCSD p-Sys	Pascal	1.1	89/07/05
AST	A	Atari ST Series	GEM	C	1.02	86/07/07
CK	B	BBN C/70	IOS 2.0	C	4E 72	89/02/02
CP	A	BigBoard II	CPM80 2.2	LASM	4.09	88/05/17
B78	D	Burroughs A Ser.	Burroughs	Algol	1.019	86/09/11
CT	C	Burroughs B20	BTOS	C	1.07	88/07/11
B68	D	Burroughs B6800	CANDE	Algol	-	85/02/15
B78	D	Burroughs B7800	Burroughs	Algol	1.019	86/09/11
B79	D	Burroughs B7900	Burroughs	Algol	5.2	85/11/27
K12	D	CESI CPU/SBC-8	OS/8 Fam.	PAL-8	10f	89/10/05
CK	B	Cadmus 68000	Unix Sys V	C	4E 72	89/02/02
CK	B	CallanUnistar300	Unix Sys V	C	4E 72	89/02/02
CD3	D	CDC Cyber	NOS	Fortran 5	3.4	88/05/10
CYB	D	CDC Cyber	NOS 2.2	Compass	1.0	86/04/17
NOS	D	CDC Cyber	NOS 2.4	Compass	1.30	87/05/19
CK	B	CDC Cyber	VX/VE5.2.1	C	4E 72	89/02/02
CDC	D	CDC Cyber 170	NOS,NOS/BE	Fortran-77	2.2	84/09/07
CIE	C	CIE 680/XX	REGULUS	C	-	87/01/26
CP	A	Cifer 1886	CPM802.2,3	LASM	4.09	88/05/17
CK	B	Codata	Unix V7	C	4E 72	89/02/02
CP	A	Comart Communica	CPM80 2.2	LASM	4.09	88/05/17
C64	A	Commodore 64	FORTH	FORTH	1.5	85/02/08
C64	A	Commodore 64/128	DOS	CROSS	2.273	89/08/10
CKI	B	Commodore Amiga	Intuition	Lattice C	4E 70	88/01/29
CP	A	Compupro IF 3/4	CPM80 2.2	LASM	4.09	88/05/17
CVK	D	Computervision	CGOS	Fortran S	1.21	87/03/04
PER	D	Concurrent 3200	OS/32	Fortran	2.1	86/09/11
PE2	D	Concurrent 3200	OS32MT72	Fortran	1.0	87/03/04
CT	C	Convergent NGEN	CTOS	C	1.07	88/07/11
CK	B	Convex C1	Unix 4xBSD	C	4E 67	87/09/14
CP	A	CPT-85xx	CPM80 2.2	LASM	4.09	88/05/17
CR	D	Cray-1,Cray-XMP	CTSS	Fortran-77	-	85/02/08
CP	A	Cromemco	CPM80 2.2	LASM	4.09	88/05/17
CN8	C	Daisy PCi	CCPM86	ASM86	2.9	86/04/10
CKD	B	Data General	AOS,AOS/VS	C	4E 70	88/01/29
RD2	D	DataGeneral 800	RDOS	BASIC	-	87/03/26
MS	A	DataGeneral DG/1	MS-DOS	MASM	3.0	90/01/16
AOS	D	DataGeneral MV	AOS,AOS/VS	SP/Pascal	-	85/02/08
AOS	D	DataGeneral S250	AOS	Fortran-5	-	84/09/14
DGM	D	DataGeneral MV	AOS/VSMVUX	C	-	85/11/27
RDO	D	DataGeneral Nova	RDOS	Fortran-5	-	84/09/14

Prefix	Tape	Machine	Operating System	Program Language	Program Version	Released yy/mm/dd
K12	D	DCC DCC-112,H	OS/8 Fam.	PAL-8	10f	89/10/05
K12	D	DEC LINC-8	OS/8 Fam.	PAL-8	10f	89/10/05
K12	D	DEC PDP-8	OS/8 Fam.	PAL-8	10f	89/10/05
K12	D	DEC PDP-8/a,e,f	OS/8 Fam.	PAL-8	10f	89/10/05
K12	D	DEC PDP-8/i,l,m	OS/8 Fam.	PAL-8	10f	89/10/05
K11	B	DEC PDP-11	IAS 3.1	Macro-11	3.60	89/06/13
MP	B	DEC PDP-11	MUMPS-11	MUMPS-1982	-	84/04/11
K11	B	DEC PDP-11	RSTS/E	Macro-11	3.60	89/06/13
K11	B	DEC PDP-11	RSX-11/M,+	Macro-11	3.60	89/06/13
K11	B	DEC PDP-11	RT-11,TSX+	Macro-11	3.60	89/06/13
RT	D	DEC PDP-11	RT-11	OMSI Pascal	2.2C	84/10/11
CK	B	DEC PDP-11	Unix 2xBSD	C	4E 72	89/02/02
CK	B	DEC PDP-11	Unix V7	C	4E 72	89/02/02
K12	D	DEC PDP-12	OS/12 Fam.	PAL-8	10f	89/10/05
K11	B	DEC Pro-3xx	P/OS	Macro-11	3.60	89/06/13
PRO	C	DEC Pro-3xx	P/OS	Bliss, Macro	1.0	84/06/01
K11	B	DEC Pro-3xx	Pro/RT	Macro-11	3.60	89/06/13
CK	B	DEC Pro-3xx	Venix V1,2	C	4E 72	89/02/02
C86	A	DEC Rainbow	CPM86	ASM86	2.9	84/12/03
MS	A	DEC Rainbow	MS-DOS	MASM	2.32A	89/01/24
QNX	C	DEC Rainbow	QNX 1.x	C	1.0	85/09/23
CK	B	DEC VAX	Ultrix-32	C	4E 72	89/02/02
CKV	B	DEC VAX	VMS	VAX-11 C	4E 72	89/02/02
VMS	B	DEC VAX	VMS	Bliss,Macro	3.3.122	90/05/15
VX	D	DEC VAX	VMS	Pascal	1.1E	84/10/11
CK	B	DEC VAX, ...	Unix 4xBSD	C	4E 72	89/02/02
MS	A	DEC VAXmate	MS-DOS	MASM	3.0	90/01/16
CP	A	DEC VT-180 Robin	CPM80 2.2	LASM	4.09	88/05/17
K12	D	DEC VT-278	OS/78 V4	PAL-8	10f	89/10/05
K12	D	DEC VT-78	OS/8 Fam.	PAL-8	10f	89/10/05
K12	D	DECmate (I)	OS/78 V4	PAL-8	10f	89/10/05
K12	D	DECmate II,III,+	OS/278 V2	PAL-8	10f	89/10/05
CP	A	DECmate-II,III	CPM80 2.2	LASM	4.09	88/05/17
MS	A	DECmate-II,III	MS-DOS	MASM	2.29	86/05/28
K10	B	DECsystem-10	TOPS-10	Bliss, Macro	3.134	89/09/18
K20	B	DECSYSTEM-20	TOPS-20	MACRO-20	4.2	88/01/25
CP	A	Delphi 100	CPM80 2.2	LASM	4.09	88/05/17
CK	B	DSI32	Unix Sys V	C	4E 72	89/02/02
CK	B	Encore Multimax	UMAX 4.2	C	4E 72	89/02/02
CP	A	Epson PX8	CPM80 2.2	LASM	4.09	88/05/17
K12	D	Fabritek MP-12	OS/8 Fam.	PAL-8	10f	89/10/05
CN8	C	Fallon 2000	CCPM86	ASM86	2.9	86/04/10
CP	A	Ferguson BigB.I	CPM80 2.2	LASM	4.09	88/05/17
CK	B	Fortune 16:32	For:Pro1.7	C	4E 72	89/02/02
CN8	C	FTS PCi	CCPM86	ASM86	2.9	86/04/10
C86	A	Fujitsu Micro16s	CPM86	ASM86	2.9	85/09/23
C86	A	Future FX20/FX30	CPM86	ASM86	2.9	86/04/10
GEC	D	GEC 4000 Series	OS4000	MUM/SERC	3.9	89/05/07
CK	B	GEC 63/40	Unix Sys V	C	4E 72	89/02/02
OS9	C	Gimex III	OS-9	C	1.5	85/09/20
CK	B	Gould PN6000	Unix 4xBSD	C	4E 72	89/02/02
CK	B	Gould PS2000	Unix Sys V	C	4E 72	89/02/02
GM2	D	Gould/SEL 32	MPX-32	Fortran 77+	2.3	86/12/10
MS	A	GRiD Compass II	MS-DOS	MASM	2.32	89/02/08
H8	D	Harris 800	VOS	Pascal,Asm	-	85/02/11
H1	D	Harris H100-1	VOS 4.1.1	Fortran-77	1.06	88/03/17
K12	D	Harris HM61x0	OS/8 Fam.	PAL-8	10f	89/10/05
CP	A	Heath H8	CPM80 2.2	LASM	4.09	88/05/17
CP	A	Heath/Zenith-89	CPM80 2.2	LASM	4.09	88/05/17
CP	A	Heath/Zenith-100	CPM85	LASM	4.09	88/05/17
MS	A	Heath/Zenith-100	MS-DOS	MASM	2.30	88/01/11
CK	B	Heurikon MiniBox	Unisoft5.2	C	4E 72	89/02/02
MU	D	Honeywell	MULTICS	PL/I	2.0h	84/09/20
DTS	D	Honeywell 6000	DTSS	Virtual PL/1	29May	86/09/16
HD6	D	Honeywell DPS6	GCOS6	C	1.0	89/04/27
CK	B	Honeywell HN XPS	UNIX V R3	C	4E 72	89/02/02
HL6	C	Honeywell L6/10	MS-DOS	MASM	1.20A	84/10/05
HDP	D	HoneywellDPS8	GCOS/TSS	B	1.1	85/03/21
HG	D	HoneywellDPS8,66	GCOS3,8	C	3.0	84/10/05
HC6	D	HoneywellDPS8,90	CP-6	PL/6	1.00	88/01/28
HCP	D	HoneywellDPS8,90	CP-6	Pascal	-	85/04/04
CK	B	HP Integral PC	HP-UX	C	4E 72	89/02/02
MS	A	HP Portable Plus	MS-DOS	MASM	2.32A	89/01/24
HP8	C	HP-86,87	HP BASIC	HP BASIC	1.01	87/04/29
MS	A	HP-110, 150	MS-DOS	MASM	2.32A	89/01/24
CP	A	HP-125	CPM80 2.2	LASM	4.09	89/08/31
HPM	D	HP-1000	RTE6,RTEA	F77 & Asm	1.99D	90/01/09
HP2	C	HP-264x	-	8080ASM	1.2	87/10/09
HP3	D	HP-3000	MPE	SPL	1.1	85/06/24
ST	D	HP-3000	SoftwTools	Ratfor	1n	84/02/18
CK	B	HP-9000 S/500	HP-UX 3.25	C	4E 72	89/02/02
HPB	C	HP-9000/200,/300	HP-BASIC	HP BASIC	1.02	89/06/21
CK	B	HP-9836 S/200	HP-UX 200B	C	4E 72	89/02/02
HP9	C	HP-9845	BASIC/SAM	HP BASIC	1.00	86/10/07
HP9	C	HP-98xx	UCSD p-Sys	HP Pascal	-	84/01/20

KERMIT VERSIONS Listed by Machine and Operating System (I-Z)

Prefix	Tape	Machine	Operating System	Program Language	Program Version	Released yy/mm/dd
MT2	D	IBM 370 Series	MTS	PLUS	-	86/11/03
MTS	D	IBM 370 Series	MTS	Assembler	-	84/01/06
MTS	D	IBM 370 Series	MTS	Pascal	1.0	84/01/06
IK	B	IBM 370 Series	MUSIC	Assembler	4.2	90/05/08
IMU	D	IBM 370 Series	MUSIC	Assembler	1.2	85/12/19
GUT	D	IBM 370 Series	MVS/GUTS	Assembler	-	85/04/05
IK	B	IBM 370 Series	MVS/TSO	Assembler	4.2	90/05/08
TS2	B	IBM 370 Series	MVS/TSO	Pascal/VS,..	2.3	87/10/01
TSN	B	IBM 370 Series	MVS/TSO	ALP/Assem	1.1A	87/09/17
IK	B	IBM 370 Series	MVSXA/TSOE	Assembler	4.2	90/05/08
CM2	B	IBM 370 Series	VM/CMS	Pascal VS	-	85/11/01
IK	B	IBM 370 Series	VM/CMS	Assembler	4.2	90/05/08
IK	B	IBM 370 Series	VM/HPO/CMS	Assembler	4.2	90/05/08
IK	B	IBM 370 Series	VM/SP3/CMS	Assembler	4.2	90/05/08
IK	B	IBM 370 Series	VM/SP4/CMS	Assembler	4.2	90/05/08
IK	B	IBM 370 Series	VM/SP5/CMS	Assembler	4.2	90/05/08
IK	B	IBM 370 Series	VM/XA/CMS	Assembler	4.2	90/05/08
CK	B	IBM 370 Series	VM/UTS 2.4	C	4E 72	89/02/02
CK	B	IBM 370 Series	VM/UTS v5	C	4E 72	89/02/02
IKX	B	IBM 370(Russian)	MVS/CICS	Assembler	2.20	89/07/10
MS	A	IBM PC,XT,AT,jr	PC-DOS	MASM	3.00	90/01/16
MS	A	IBM PC Portable	PC-DOS	MASM	3.00	90/01/16
MS	A	IBM PC Convertib	PC-DOS	MASM	3.00	90/01/16
QK	A	IBM PC,XT,AT	PC-DOS	Turbo Pascal	3.1	88/12/14
CK	B	IBM PC/AT	Xenix/286	C	4E 72	89/02/02
CK	B	IBM PC/XT	PC/IX	C	4E 72	89/02/02
PIC	D	IBM PC/XT,AT	PICK	DATA/BASIC	0.3	89/08/21
MS	A	IBM PS/2 Series	PC-DOS	MASM	3.00	90/01/16
CKO	B	IBM PS/2 & AT	OS/2	C	1.0p	89/03/15
UCI	C	IBM PC family	UCSD p-Sys	UCSD Pascal	0.1	84/05/23
MX	B	IBM PC family	MINIX	C	4D 61	88/05/17
TP4	A	IBM PC family	PC-DOS	Turbo Pascal	1.1a	88/04/15
WK	C	IBM PC family	PC-DOS	Lattice C	1.18	86/05/22
K12	D	IBM PC w/EMUL8	OS/8 Fam.	PAL-8	10f	89/10/05
CK	B	IBM RT PC	ACIS 4.2	C	4E 72	89/02/02
CK	B	IBM RT PC	AIX	C	4E 72	89/02/02
VME	D	ICL 2900	VME	S3	1.01	87/07/14
CN8	C	ICL PC 2,Quattro	CCPM86	ASM86	2.9	87/05/17
PQK	C	ICL/Perq	Perq OS	Pascal	2.0	84/12/04
PQ2	C	ICL/Perq	POS D6/R2	POS Pascal	-	85/06/24
MS	A	Intel 300 Series	iRMX-286	MASM/ASM86	2.30	88/05/02
MS	A	Intel 300 Series	iRMX-86	MASM/ASM86	2.30	88/05/02
CK	B	Intel 310	Xenix 3.0	C	4E 72	89/02/02
RMX	C	Intel 86,286	RMX 1.0	PL/M	1.0	85/10/25
I86	C	Intel 86/380	iRMX-86	PL/M	2.3	85/09/23
IRM	C	Intel 86/380	iRMX-86	PL/M	2.41	87/03/04
MD2	C	Intel MDS	ISIS	PL/M	-	85/11/01
MDS	C	Intel MDS	ISIS	PL/M	-	87/04/06
K12	D	Intersil IM6100	OS/8 Fam.	PAL-8	10f	89/10/05
K12	D	IntersilIntercept	OS/8 Fam.	PAL-8	10f	89/10/05
CP	A	Ithaca Intersys	CPM80 2.2	LASM	4.09	88/05/17
UCJ	C	J LoeblMagiscan2	UCSD p-Sys	UCSD Pascal	-	86/06/23
CP	A	Kaypro 4	CPM80 2.2	LASM	4.09	88/05/17
CP	A	Kaypro II	CPM80 2.2	LASM	4.09	88/05/17
M2	C	Lilith Worksta.	Medos	Modula-2	1.0	87/05/17
LM	C	LMI Lispmachine	LMI-LAMBDA	ZETALISP	1.0	85/09/12
CP	A	Lobo Max-80	CPM80 2.2	LASM	4.09	88/05/17
LUX	C	Luxor ABC-80	ABC-DOS	Z80 Asm	1.0	89/12/12
LUX	C	Luxor ABC-800	ABC/UFDDOS	ABC-BASIC-II	4.10	89/12/12
MS	A	Macintosh/AST286	MS-DOS	MASM	3.0	90/01/16
MBF	D	MAI Basic Four	BOSS/VS	BASIC BB86	1.0	88/04/11
CK	B	Masscomp	RTU 2.2	C	4E 72	89/02/02
CP	A	Merlin M2215	CPM80 2.2	LASM	4.09	88/05/17
PIC	D	MicroDataREALITY	PICK	DATA/BASIC	0.3	89/08/21
CP	A	Micromint SB180	CPM80 2.2	LASM	4.09	88/05/17
MOD	D	MODCOMP Classic	MAX IV	Fortran/ASM	A.0	87/01/26
CP	A	MorrowDecisionI	CPM80 2.2	LASM	4.09	88/05/17
CP	A	MorrowMicroDec.I	CPM80 2.2	LASM	4.09	88/05/17
CK	B	Motorola 4 Phase	Unix Sys V	C	4E 72	89/02/02
FL	C	Motorola 6809	Flex	Assembler	-	86/02/14
FL2	C	Motorola 6809	FLEX-09	C	3.0	87/03/04
WIN	A	MicrosoftWindows	MS/PC-DOS	Microsoft C	4.11	90/01/19
CP	A	NCR Decisionmate	CPM80 2.2	LASM	4.09	88/05/17
CK	B	NCR Tower	OS 1.02	C	4E 72	89/02/02
CK	B	NCR Tower	Unix Sys V	C	4E 72	89/02/02
ND	D	ND-10/100/500	SintranIII	ND-Pascal	3.1b	85/06/24
C86	A	NEC APC	CPM86	ASM86	2.9	84/12/03
MS	A	NEC APC	MS-DOS	MASM	2.29	86/05/28
MS	A	NEC APC III	MS-DOS	MASM	2.30	88/03/21
CK	B	NEC Astra XL	UNIX	C	4E 72	89/02/02
MS	A	NEC PC9801	MS-DOS	MASM	2.32A	89/04/27
CP	A	Nokia MikroMikko	CPM80 2.2	LASM	4.09	88/05/17
CP	A	NorthstarAdvanta	CPM80 2.2	LASM	4.09	88/05/17
CP	A	NorthstarHorizon	CPM80 2.2	LASM	4.09	88/05/17
CP	A	Ohio Scientific	CPM80 2.2	LASM	4.09	88/05/17
MS	A	Olivetti M24 PC	MS-DOS	MASM	3.0	90/01/16
CN8	C	Orion PCi	CCPM86	ASM86	2.9	86/04/10
K6	A	OS9/68K Portable	various	Assembler	1.0	87/07/22
CP	A	Osborne 1	CPM80 2.2	LASM	4.09	88/05/17
UCM	C	PascalMicroengin	UCSD p-Sys	Pascal	III.0	84/12/03
QNX	C	IBM PC,XT,AT	QNX 1.x	C	1.0	85/09/23
K12	D	PCM PCM-12	OS/8 Fam.	PAL-8	10f	89/10/05
PER	D	PerkinElmer 3200	OS/32	Fortran	2.1	86/09/11
CK	B	PerkinElmer 3200	Unix V7	C	4E 72	89/02/02
CK	B	PerkinElmer 3200	Xelos	C	4E 72	89/02/02
PE7	D	PerkinElmer 7000	IDRIS	C	1.1 0	86/12/08
CK	B	PerkinElmer 7350	Uniplus	C	4E 72	89/02/02
CK	B	Plexus P/35,P/60	Unix Sys 3	C	4E 72	89/02/02
CP	A	PMC Micromate101	CPM80 2.2	LASM	4.09	88/05/17
PRI	D	Prime	PRIMOS	PL/P	8.00	90/01/19
CK	B	Pyramid 90x	Unix 4xBSD	C	4E 72	89/02/02
CP	A	Rair Black Box	CPM80 2.2	LASM	4.09	88/05/17
CK	B	Ridge 32	ROS 3.2	C	4E 72	89/02/02
CP	A	RM380ZF	CPM80 2.2	LASM	4.09	88/05/17
CP	A	RM380ZM	CPM80 2.2	LASM	4.09	88/05/17
RM	C	RML 480Z	ROS 2.x	C	1.22	86/11/03
RM	C	RML Nimbus	MS-DOS?	C	1.22	86/11/03
CP	A	Sanyo 1100 MBC	CPM80 2.2	LASM	4.09	88/05/17
MS	A	Sanyo 550 MBC	MS-DOS	MASM	2.30	88/05/16
CP	A	ScreenTyper	CPM80 2.2	LASM	4.09	88/05/17
MS	A	Seequa Chameleon	MS-DOS	MASM	2.32A	89/01/24
CK	B	Sequent Balance	DYNIX 2,3	C	4E 72	89/02/02
CK	B	Sequent Symmetry	DYNIX 3.0	C	4E 72	89/02/02
QL2	C	Sinclair QL	QDOS	BCPL	-	87/05/15
QLK	C	Sinclair QL	QDOS	C	1.10	87/05/15
UN	D	Sperry 1100	Exec	Assembler	2.5	86/09/03
UN	D	Sperry 1100	Exec	NOSC Pascal	2.0	84/10/08
ST	D	Sperry 1100	SoftwTools	Ratfor	1n	84/02/18
SP9	D	Sperry 90/60	VS9	Assembler	-	86/04/09
CK	B	SUN Microsystems	SUNOS 3.x	C	4E 72	89/02/02
CK	B	SUN Microsystems	SUNOS 4.x	C	4E 72	89/02/02
CP	A	Superbrain	CPM80 2.2	LASM	4.09	88/05/17
LM	C	Symbolics 36xx	Lisp	ZETALISP	1.0	85/09/12
TAN	D	Tandem Nonstop	Guardian	TAL	1.0	86/04/08
TA2	A	Tandy 2000	MS-DOS	MASM	1.20	84/02/16
TU	E	Tape Utilities	(various)	(various)	-	88/06/06
C86	A	Tektronix 4170	CPM86	ASM86	2.9	84/12/03
CP	A	Telcon Zorba	CPM80 2.2	LASM	4.09	88/05/17
CP	A	Teletek Skymastr	CPM80 2.2	LASM	4.09	88/05/17
CP	A	Teletek/ADM-22	CPM80 2.2	LASM	4.09	88/05/17
UCT	C	Terak	UCSD p-Sys	UCSD Pascal	-	84/04/11
EXP	C	TI Explorer	LISP	Common Lisp	1.0	87/03/04
CK	B	TI NU-Machine	Unix Sys V	C	4E 72	89/02/02
CK	B	TI Pro/Opus32016	Unix Sys V	C	4E 72	89/02/02
MS	A	TI Professional	MS-DOS	MASM	2.29	86/05/28
TI9	D	TI 990	DX10	Pascal	1.0	87/07/10
CP	A	Torch Unicorn 5	CPM80 2.2	LASM	4.09	88/05/17
K12	D	TPA (Hungary)	OS/8 Fam.	PAL-8	10f	89/10/05
CC	A	TRS-80 CoCo	DOS	EDTASM	1.1	85/03/21
OS9	C	TRS-80 CoCo	OS-9	C	1.5	85/09/20
TRS	A	TRS-80 I and III	TRSDOS	M80	3.5	84/08/08
CK	B	TRS-80 Model 16	Xenix	C	4E 72	89/02/02
CP	A	TRS-80 Model 4	CPM80 2.2	LASM	4.09	88/05/17
M4	A	TRS-80 Model 4	TRSDOS	ASM	5.2	86/10/29
CP	A	TRS-80 Model II	CPM80 2.25	LASM	4.09	88/05/17
TR2	A	TRS-80 Model II	TRSDOS	Assembler	1.2	87/03/26
UM	C	UMicro U-MAN1000	CP/M-68K	C and Asm	-	86/04/10
CP	A	USmicroSalesS100	CPM80 2.2	LASM	4.09	88/05/17
CK	B	Valid Scaldstar	Unix 4xBSD	C	4E 72	89/02/02
PIC	D	VAX/Ultimate	PICK	DATA/BASIC	0.3	89/08/21
CP	A	Vector Graphics	CPM80 2.2	LASM	4.09	88/05/17
C86	A	Victor/Sirius 1	CPM86	ASM86	2.9	86/07/07
MS	A	Victor/Sirius 1	MS-DOS	MASM	2.32A	89/01/24
VIC	C	Victor/Sirius 1	MS-DOS	C	1.0	84/09/07
CP	A	Video Genie	CPM80 2.2	LASM	4.09	88/05/17
MS	A	Wang PC/APC	MS-DOS	MASM	2.31	88/08/13
CK	B	Whitechapel MG-1	Genix-1.3	C	4E 72	89/02/02
CP	A	Xerox 820	CPM80 2.2	LASM	4.09	88/05/17
CP	A	Z80MUdevelpmtSys	CPM80 2.2	LASM	4.09	88/05/17
CK	B	Zilog Sys 8000	Zeus 3.20	C	4E 72	89/02/02
KU	E	Kermit User Guide		English	6thEd	89/03/06
KP	E	Kermit Protocol Manual		English	6thEd	86/06/01
IMA	E	Info-Kermit Digests		English		85-now
NEW	E	Kermit News Issues		English		86-now
BYT	E	BYTE Mag Kermit Article		English		84/06/01
DC	E	Data Comm Mag Articles		English		87-now
GER	E	Introduction to Kermit		German		87/10/08
POR	E	Kermit User Guide		Portuguese	5thEd	87/09/17
POR	E	Kermit Protocol Manual		Portuguese	5thEd	87/09/17

Donors

Thanks to the following people from all over the world who checked the Contribution box on the Kermit order form. Their contributions help to keep the Kermit effort alive.

Riyadh Alsharekh, Kenny Alvarez, Barbara Bailey, Scott Bailey, Mary Barry, Stephen Bean, Patricia Beard, Thomas Bing, Peter Bjorklund, Donald Blum, Bob Booth, Barry Brevik, Dennis Broeckel, Usama Bsata, Thomas Burnside, Sal Buzzetta, Edwin Calka, Robert Chaput, Dale Chatham, M.J. Chichester, Marty Cole, Andrew Comas, Ruben Conti, R. Coppola, Harold Corwin, Tony Cosentino, Robert Craig, Americo Lourenco da Silva, Tavarg da Silva, Ivy Dang, Delaine Davis, Guy DeStefano, G. Devaney, James Drumgole, Barry Elkin, Frank Fedele, R.B. Folsom, James Frazee, Richard Fry, Zbigniew Fryzlewicz, Andrew Gaeta, John Galloway, Gary Goodman, Ernst Habicht, Carlos Hannessian, Paul Hardiman, Toby Heaton, Wilfried Heinickel, Patrick Helbach, Cliff Hemming, Art Herbert, Tim Hiebert, Paul Higgins, Don Hughes, William Hunt, James Hurley, Doug Johnson, Michael Jokich, S.J. Landry, Jr., S. Kiwic, Roger Knake, Ken Knevel, Peggy Knox, Stephen Lacovette, Arthur Layzer, Helena Lee, Scott Lindall, Lon-Mu Liu, Robert Loepp, Kathy Martin, Henry Matelson, D. McDonnell, Robert McKelvey, Laura McWilliams, L. Miller, Mary Miller, C.A. Monley, Saul Newman, Ed O'Connell, Ben Palmer, M. Parkhideh, Michael Pearson, Norman Pickup, Cesar Pinto, Cal Powell, Rob Rasins, M. Rasley, Michael Rawlins, Paul Rea, A. Reeve, Bonnie Resse, Glenn Rhoades, George Roberts, Vicki Robins, Don Ross, Anna Rubin, Frank Rubin, Walter Rue, Martin Schmidt, Barry Scott, John Sloan, Harry Smith, Robert Strickland, Michael Sundermann, L.P. Swanson, Raleen Tautfest, Bob Temple, Diane Thoman, Brian Wallace, John Wallace, Bob Weakley, Nurit Weiss, Bob White, Robert Winnewisser, and Steve Wright.

And thanks to the following corporations, agencies, and universities that donated funds to the Kermit effort in 1988-90:

Alberta Public Works (Canada), Crain/Atlanta Inc, DECUS Japan, DECUS Switzerland, Digital Computer of Japan, Fujitsu Systems of America Inc, General Electric Co, Hewlett Packard GmBH (West Germany), Martin Marietta, Meridian Technology Corp, Merrill Lynch, Mobile Oil, Motorola Inc, NCR GmbH (West Germany), Nippon Telephone and Telegraph (Japan), Sandoz Pharmaceuticals Corp, Université du Québec à Chicoutimi (Canada), and Zhejiang University (Peoples Republic of China).

Date Ordered: ______/______/______ #________________

PLEASE NOTE: Prices, terms and items are subject to change. Before you order, check that the form is not dated more than 6 months prior. If so, please write to Kermit Distribution to request a new order form. Also, please order carefully since we **CANNOT** refund or exchange items. **Orders are normally processed within 2-4 weeks.** Prices are in US dollars ($), first price for North America / second for outside North America. Please pay by check.

9-TRACK MAGNETIC TAPE. All Kermit source included. Each tape order comes with printed copies of the Kermit User Guide and Protocol Manual. Price: $120 / $155 PER TAPE. Check each desired tape:

Tape / Format:	ANSI	TAR	OS
Tape A (popular micros):	❒	❒	❒
Tape B (popular mainframes):	❒	❒	❒
Tape C (other micros):	❒	❒	❒
Tape D (other mainframes):	❒	❒	❒
Tape E (documentation):	❒	❒	❒

❒ PRIME computers: specify ANSI. Check here for free listing of an ANSI-tape reader program in Fortran.

TAPE SUBTOTAL ($120 / $155 times number of tapes) . . . . $____________

TAPE CARTRIDGES: $250 / $280 EACH, manuals included:

Tape / Format:	TKV	TKU	SUN
Contents of Tapes A, B, and E:	❒	❒	❒
Contents of Tapes C, D, and E:	❒	❒	❒

TAPE CARTRIDGE SUBTOTAL ($250 / $280 times number of cartridges) . . . . $____________

IBM PC/PS2 MS-DOS Kermit executable program diskette with book *Using MS-DOS Kermit:*

❒ Book + 5.25-inch 360K DOS diskette: $29.95 / $40 . . . . $____________
❒ Book + 5.25- *and* 3.5-inch DOS 720K diskette: $39.95 / $50 . . . . $____________

IBM PC/PS2 MS-DOS Kermit utilities & technical documentation diskettes:

❒ Two 5.25-inch diskettes: $20 / $26 . . . . $____________
❒ Two 3.5-inch diskettes: $20 / $26 . . . . $____________

Other Kermit programs on diskette, manual included: $20 / $25 EACH:

❒ *Apple Macintosh* (3.5-inch Mac SS diskette) . . . . $____________
❒ *DEC MicroVAX; VMS* (5.25-inch RX50) . . . . $____________
❒ *DEC MicroVAX; Ultrix* (5.25-inch RX50) . . . . $____________
❒ *OS/2 Kermit Beta Test* (3.5-inch DOS diskette) . . . . $____________

Kermit source code on diskettes:

❒ MS-DOS IBM PC/PS2 Kermit MASM 5.0 Sources, 4 DOS 5.25" 360K disks: $40 / $47 . $____________
❒ UNIX C-Kermit C-Language Sources, 2 DOS 5.25" disks + manual: $30 / $35 . . . . $____________
❒ Macintosh Kermit MPW C Sources, 1 Mac 3.5" disk + manual: $20 / $25 . . . . $____________

Printed documents, enter quantity:

❒ Book: *Kermit, A File Transfer Protocol:* $28 / $43 . . . . $____________
❒ Manual: *Kermit User Guide:* $20 / $35 . . . . $____________
❒ Manual: *Kermit Protocol Manual:* $10 / $15 . . . . $____________
❒ Separate chapters from the Kermit User Guide: $10 / $13 each:
❒ UNIX Kermit ❒ VMS Kermit ❒ Macintosh Kermit . . . . $____________
❒ *BYTE* Magazine article manuscript, 1984: $5 / $8 . . . . $____________
❒ *Data Communications* Magazine software packages article manuscript, 1987: $5 / $7 . $____________
❒ *Data Communications* Magazine digital PBX article manuscript, 1988: $5 / $7 . . . . . . $____________
❒ International Character Set Proposal: $5 / $7 . . . . $____________
❒ Info-Kermit Digest Volumes, paginated & indexed: $15 / $25 each:
❒ V3 1985 ❒ V4-5 ’86 ❒ V6 ’87 . . . . $____________
❒ V7-8 1988 ❒ V9-10 ’89 ❒ V11-12 ’90 . . . . $____________

SIDE 1 SUBTOTAL (please complete side 2 also) . . . . $____________

SUBTOTAL FROM SIDE 1 . . . . $____________

❒ ORDER PROCESSING FEE: If you can NOT prepay with a check, please include **BOTH**:

1. A $100.00 Order Processing (Billing) Fee: . . . . $____________
2. **AND** a Purchase Order (terms net 30, FOB origin):

Purchase Order Number: ______________________________
(not required if you prepay by check.)

OVERSEAS SHIPMENT (outside of USA, Canada, and Mexico):
Method 1, Airmail: Pay second price shown for each item.
Method 2: Call (USA) 212-854-3703 for alternate shipping arrangements.

❒ USA RUSH ORDERS (Priority handling, Federal Express), add $30.00 . . . . $____________

❒ VOLUNTARY TAX-DEDUCTIBLE CONTRIBUTION . . . . $____________
Please support the Kermit development and distribution effort.

GRAND TOTAL, SIDES 1 AND 2 *(Do Not Add Sales Tax)* : . . . . $____________ (=)

IMPORTANT: WRITE YOUR SHIPPING ADDRESS HERE, *exactly as it should appear on mailing label*. Please don’t use a Post Office Box number, because UPS or other delivery services cannot deliver to PO Boxes.

Name: ______________________ Organization: ______________________

Address: ____

City: ______________ State or Province: __________ Zip or Postal Code: ______________

Country: ______________________ Phone: ______________________

MAIL COMPLETED ORDER FORM TO:

Kermit Distribution, Dept. OP
Columbia University Center for Computing Activities
612 West 115th Street
New York, NY 10025 (USA)

MAKE CHECKS IN US DOLLARS PAYABLE TO:

Columbia University Center for Computing Activities

CUCCA USE ONLY:

Received: ______________ Logged: ______________ By: __________

Shipped: ______________ Via: ______________ By: __________

Shipper’s Reference: ______________________________________

Directory

Postal Address:

Kermit Development and Distribution
Columbia University Center for Computing Activities
612 West 115th Street
New York, NY 10025 USA

Telephone:

Ordering Information:	(USA) 212-854-3703
Technical Support:	(USA) 212-854-5126

Networks:

BITNET/EARN:	`KERMIT@CUVMA`
Internet:	`Info-Kermit@watsun.cc.columbia.edu`

Kermit Program Contributors

A large number of individuals and organizations have contributed Kermit programs, enhancements, bug fixes, and documentation to Columbia University's Kermit Distribution. Thanks to each and every contributor. Space does not permit listing the names all of the people who have given their time to the Kermit effort over the years, but here are just some of the institutions with which they are affiliated:

Australia: *University of New South Wales, CSIRO.* ***Belgium:*** *University of Liège, University of Namur.* ***Brazil:*** *Telecomunicações Brasileiras SA.* ***Canada:*** *Queen's University, McGill University, University of British Columbia, University of Toronto, Victor Canada.* ***France:*** *CiSi télématique, INFOGEM.* ***Finland:*** *Helskini University of Technology.* ***Hungary:*** *Central Research Institute for Physics.* ***Ireland:*** *University College Dublin.* ***Italy:*** *Bologna University, SGS Systems Division.* ***Japan:*** *National Lab for High-Energy Physics, Nippon Telephone and Telegraph Research Labs.* ***Netherlands:*** *Rijksuniversiteit Groningen, Technische Hogeschool Eindhoven, Vrije Universiteit; Philips International.* ***Norway:*** *University of Oslo, University of Trondheim, Norwegian Institute of Technology, Norsk Hydro Data.* ***Sweden:*** *ABC-Klubben, Central Bank of Sweden, Gothenburg University, Stacken Computer Club, Stockholm University Computer Centre (QZ), Peridot Konsult, University of Wasa.* ***Switzerland:*** *University of Bern, University of Zürich, Ecofin Research and Consulting, Ltd.* ***United Kingdom:*** *Aberdeen University, Bath University, Birmingham University, Brighton Polytech, Brunel University, Edinburgh University, Imperial College, Lancaster University, Liverpool University, Loughborough University, Nottingham University, Salford University, Sheffield City Polytechnic, University College London, University of Manchester, University of York; Acorn Computers, Rutherford Appleton Laboratory.* ***USA:*** *Brigham Young University, Brown University, Bucknell University, California State University, Carnegie-Mellon University, Cerritos College, Clemson University, Colorado School of Mines, Columbia University, Cornell University, Dartmouth University, Grinnell College, Harvard University, Harvard Smithsonian Center for Astrophysics, Indiana/Purdue University, Lehigh University, Louisiana State University, Ohio State University, McGill University, Massachusetts Institute of Technology, Oakland University, Oklahoma State University, Rice University, Rutgers University, Stevens Institute of Technology; the Universities of: Arizona, California (many branches), Hawaii, Kansas, Maryland, Michigan, North Carolina, Rochester, Southern California, Tennessee, Texas, Toledo, Vermont, Virginia, Washington, Wisconsin; Utah State University, US Military Academy, Virginia Polytechnic Institute, Worcester Polytech Institute; Advanced Computer Communications, American Mathematical Society, American Telephone and Telegraph, AT&T Bell Laboratories, Atari Computer, Bonneville Power Authority, Brookhaven National Laboratory, Cantor Consulting, Digital Equipment Corporation, Diversified Computer Systems, DuPont Co, Eastman Kodak, E-Systems Inc, Environmental Protection Agency, Hewlett Packard Co, Honeywell Information Systems, Hughes Aircraft, Intel Corp, Johnson Control, Leigh Instruments, Lisp Machines Inc., Los Alamos National Lab, Marconi Space, Merrell-Dow, Metacomco Ltd., National Institutes of Health, National Aeronautics and Space Administration, Naval Ocean Systems Center, Perkin-Elmer, Planning Research Corporation, Polaris Inc, Prime Computer, Quest Research, RCA Labs, RTI, SAS Institute, Simulation Associates, Soft Machines, Sphere Holdings, SPSS Inc., The Source Telecomputing, TransEra Corp., Tridom Corp., UNISYS Corp., US Navy, WRIST Inc.* ***USSR:*** *International Centre for Scientific and Technical Information, Moscow;* ***West Germany:*** *Wissenschaftliches Institut der Ortskrankenkassen Bonn, Technical University of Berlin, Technical University of Darmstadt, University of Würzburg.*

Kermit News

Kermit Development and Distribution
Columbia University Center for Computing Activities
612 West 115th Street
New York, NY 10025 USA

Non Profit Org.
U.S. Postage
PAID
New York, N.Y.
Permit # 3593