UNET -- Standard Communication Protocols for UNIX

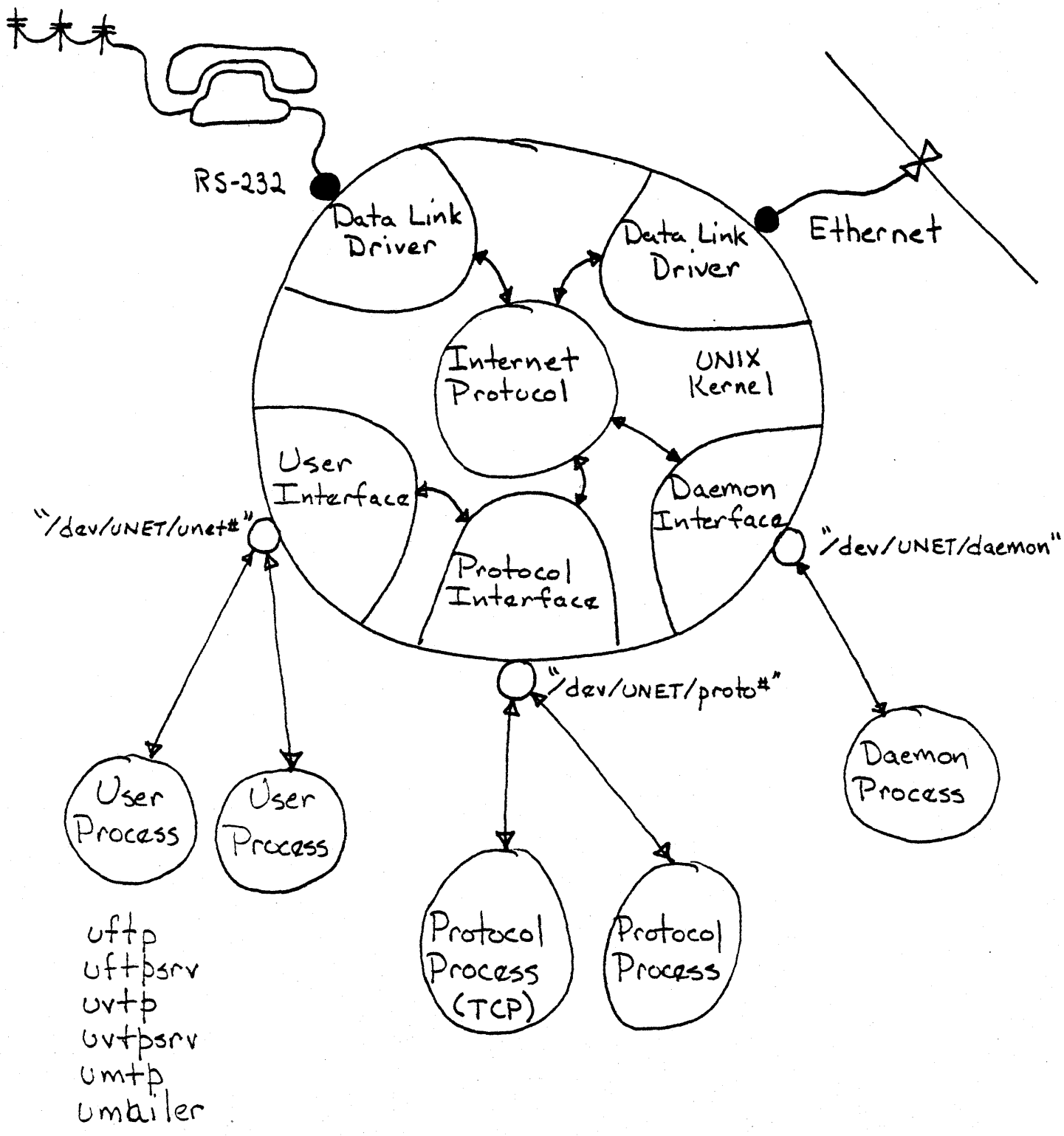
A seminar presented by 3Com Corporation

11 November 1980

Bruce S. Borden
Howard S. Charney
Robert M. Metcalfe
Gregory L. Shaw

3Com Corporation
3000 Sand Hill Road #1
Menlo Park, California 94025
(415) 854-3833

UNET Architecture



UNET IS A TRADEMARK OF 3COM CORPORATION

3COM'S UNET SOFTWARE PROVIDES COMMUNICATION AMONG COMPUTERS RUNNING BELL'S UNIX OPERATING SYSTEM.

UNET TRANSFERS FILES, CONNECTS REMOTE TERMINALS, AND CARRIES ELECTRONIC MAIL USING DOD'S STANDARD INTERNET PROTOCOL (IP) AND TRANSMISSION CONTROL PROTOCOL (TCP). UNET ALSO OFFERS GENERAL INTERPROCESS COMMUNICATION FACILITIES TO UNIX PROGRAMMERS.

UNET CREATES AN IP/TCP PACKET INTERNET AMONG COMMUNICATING UNIX COMPUTERS ALLOWING ITS FILE, TERMINAL, MAIL, AND PROGRAM SERVICES TO TRAVERSE A VARIETY OF COMMUNICATION MEDIA BOTH LOCAL AND REMOTE. SUCH MEDIA INCLUDE BELL SYSTEM VOICE CIRCUITS, DIGITAL SATELLITE CHANNELS, AND ETHERNET.

UNET WILL ENHANCE BELL'S UNIX ON PDP-11S AND VAXS AND IS BEING OFFERED ON UNIX WHEN PORTED TO OTHER MACHINES. IN PARTICULAR, UNET WILL BE AVAILABLE WITH XENIX ON THE I8086, Z8000, AND M68000 IN SPECIAL COOPERATION WITH MICROSOFT.

UNET Seminar Schedule

(1) Bob Metcalfe	3:00	Where UNET fits in.
(2) Howard Charney	3:25	How to buy UNET licenses.
(3) Greg Shaw	3:50	UNET features and functions.
(4) Break	4:15	Refreshments.
(5) Bruce Borden	4:30	UNET internals and UNIX.
(6) Questions	5:30	Discussion.
(7) End	6:00	End.

(1) Bob Metcalfe will present 3Com's view of where IP/TCP and UNET fit into ISO's OSI Reference Model, CCITT's X.25, IBM's SNA, Xerox's Ethernet, and 3Com plans for building computer communication compatibility.

(2) Howard Charney will present how 3Com is offering UNET for wide distribution and how you can buy licenses to obtain UNET sources, installation, maintenance, and binaries for resale.

(3) Greg Shaw will present UNET's features and functions and will explain how UNET is used with files, terminals, mail, and programs.

(4) Bruce Borden will present UNET's internals and how UNET is implemented in harmony with UNIX.

3Com stands for Computer Communication Compatibility

3Com Corporation founded in June 1979.

Packet switching and local computer networks.

3Com people: Paul Baran, Director
Bruce S. Borden, UNET Product Manager
Howard S. Charney, VP Operations and Counsel
Ronald C. Crane, Director of Engineering
Robert M. Metcalfe, President
Kenneth P. Morse, VP Marketing and Planning
Gregory L. Shaw, VP Software Engineering
David A. Spiller, Director of Finance and Treasurer

3Com's four products:

Consulting and contract product development
The 3Com Local Computer Network Vendor List
UNET, IP/TCP for UNIX
DEC Intel 3Com Xerox Ethernet

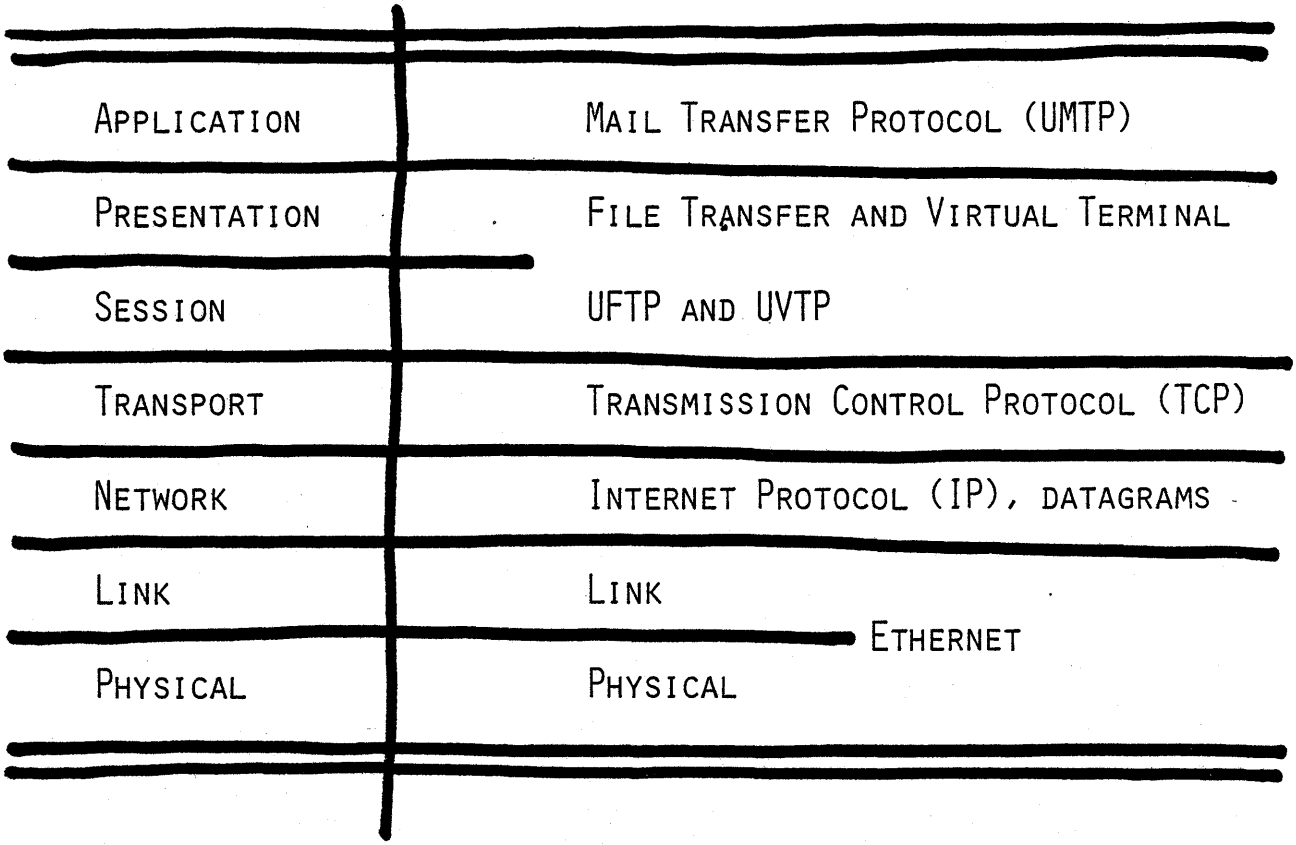
IP/TCP is as close as you can get to ISO's OSI Reference Model

US DOD Internet Protocol and Transmission Control Protocol

Internation Standards Organization's Open Systems Interconnection Reference Model

Multivendor layered packet internetting

Not exactly, but as close as you can get:



CCITT's X.25 and IP/TCP and Ethernet

X.25 is an international standard interface for connecting data terminal equipment (DTE) to data communication equipment (DCE). X.25 specifies physical, link, and network level interfaces for packet exchanges that create, use, and close virtual circuits between subscriber DTEs through PTT DCEs.

IP/TCP is a US DOD standard end-to-end protocol through which communicating computers exchange internet datagrams and through which virtual circuits can be maintained between processes on communicating computers. The DCE through which IP/TCP internet packets travel may be either public or private, remote or local, active or passive.

Ethernet is a standard now being promoted by DEC, Intel, 3Com, and Xerox. It specifies both physical and link level interconnection. Ethernet can serve to carry internet datagrams for IP/TCP, as can X.25. When two DTEs are locally connected by Ethernet, there is no DCE to which they can interface using X.25 (and HDLC and X.21).

IP/TCP, Ethernet, and SNA are not alternatives.

Common inferences from Xerox Information Outlet advertisements notwithstanding, Ethernet is not better (or even worse) than IBM's System Network Architecture (SNA).

SNA has been developed by IBM to tackle the whole problem of data communication and can be likened to ISO's OSI Reference Model. SNA subsumes functions like those provided by IP/TCP and Ethernet.

IP/TCP is a non-vendor-aligned protocol that might be used instead of certain of SNA's functions. IP/TCP emphasizes distributed peer communication while maximizing end-to-end coordination. SNA emphasizes host-centered communication while maximizing centralized control.

Ethernet is a proposed non-vendor-aligned standard local computer networking system that might be used instead of certain of SNA's functions. For example, Ethernet might be used in an SNA system instead of SDLC and X.21. Like IP/TCP, Ethernet emphasizes distributed peer communication. Where SDLC and X.21 presume the existance of central controllers and separate DCEs, Ethernet is distributed and passive having no central controllers or separate DCEs.

3Com's approach is to invest in forward-looking standards and to maximize compatibility.

Today's versus tomorrow's installed base.

The big three problems:

Engineers innovating incompatibility

Marketeers differentiating products

The Trap: making standards the hard way

Standards chosen so far:

DEC Intel 3Com Xerox Ethernet (IEEE?)

DOD IP/TCP (waiting for ISO) + Arpanet

UNIX and C (for the 16-bit micros)

Standards being studied:

I8086, Z8000, M68000

X.25

FTP, VTP, MTP (files, terminals, mail)

Fiberoptics

Database Manager

Document Format

DEC Intel 3Com Xerox Ethernet

Critical mass has been achieved.

Xerox 860 and 5700. Other products?

DEC transceivers and controllers. When?

Intel chip(s). What and when?

Other participants? HP,.....?

3Com: LCNVL, UNET, transceivers, controllers, ... soon.

Ethernet is only 1% or 2% of the problem.

IP/TCP and UFTP, UVTP, and UMTP are after the other 98%.

UNET Standard Communication Software for UNIX

- Licensing of UNET parallels UNIX.
- The four types of UNET licenses.
- The UNET Licensing fees.
- Obtaining UNET.

presented by Howard Charney
November 11, 1980

Why do people license UNIX?

- Technically superior OS; tools

- UNIX is written in C, a high level language.

- Applications programs (i.e. portability of C and compatibility)

- The cost of licensing is far less than the cost of recreating the code (including the documentation) if you had the expertise and the time to do it.

Why should you license UNET?

- IP/TCP are technically superior protocols.

- UNET is written in C, a high level language.

- IP/TCP are DOD standards and are non-vendor aligned.

- The cost of licensing UNET is far less than the cost of recreating the code (including UFTP, UVTP, UMTP, and the documentation) <u>if</u> you had the expertise <u>and</u> the time to do it.

UNIX is supplied

- as source code from Western Electric after you execute the UNIX Software Agreement.
- without installation.
- without maintenance.
- without warranties of any kind.
- as object code from Western Electric Licensees* who execute a Supplemental Agreement.

* and soon from our friends at Microsoft as their XENIX OS.

UNET is supplied

o as source code from 3Com Corporation after you execute the UNET Software Agreement.

o with installation support -- if you decide you need installation (i.e. dropping UNET into the version 7 kernel, writing network drivers, etc.) and you execute a UNET Installation Agreement.

o with maintenance support -- if you decide you need maintenance (i.e. bug fixes, documentation updates, etc.) and you execute a UNET Maintenance Agreement.

o with warranties -- that 3Com owns UNET and that UNET works as we represent.

o as object code from 3Com licensees who execute a UNET Supplemental Agreement.

Thus, there are four UNET licenses:

- right-to-use source code (i.e. Software Agreement*).
- installation (i.e. Installation Agreement).
- maintenance (i.e. Maintenance Agreement).
- right-to-sell binaries (i.e. Supplemental Agreement*).

* parallels Western Electric UNIX license.

RIGHT-TO-USE SOURCE LICENSE

- $7,300 for a first CPU.
- $4,300 for each additional CPU.

INSTALLATION

- is quoted individually based upon each customer's needs.

- is performed on a consulting-like basis including expenses.

- requires an advance payment to 3Com of approximately one-half of the estimated total fees for the installation.

MAINTENANCE

- is calculated as 5% of the sum total of UNET license fees paid to 3Com for source and binary licenses.

- is paid every six months.

RIGHT-TO-SELL BINARIES LICENSE

- Fees parallel UNIX.

- Fees are calculated from a table based upon the number of users per CPU and the total license fees paid to 3Com.

- Fees are approximately one-third of the comparable UNIX fees.

20K RITE TO SELL BINARIES

AVAILABILITY OF UNET

- UNET is available for shipment to customers on DECEMBER 15, 1980.

- If you desire installation, please so indicate to 3Com early, because 3Com personnel resources are a gaiting factor.

HOW DO YOU OBTAIN UNET?

- You fill out a request indicating licenses desired, model number, serial number, and location of CPU's, etc.

- If installation is desired, please specify preferred dates.

- 3Com prepares license agreement(s) [please allow two weeks].

- You execute the license agreement(s) and make appropriate payment to 3Com Corporation.

- 3Com prepares a tape and documentation and ships them to you [please allow three days at 3Com and three days for shipment to you].

WHAT UNET DOES

General Characteristics

- o communication software for UNIX Version 7
- o uses DoD Standard IP/TCP protocols
- o written in C
- o supports any physical network medium
- o drop-in installation

Programs

- o UFTP (UNET File Transfer Program)
  - o interactive or command line mode
  - o file functions: get, put, delete, rename
  - o directory functions: list, change
  - o automatic login
  - o remote !
- o UVTP (UNET Virtual Terminal Program)
  - o complete remote terminal emulation
  - o local !
- o UMTP (UNET Mail Transfer Program)
  - o functions: send, read
  - o mail header fields: to, from, date, subject, cc
  - o compatible with RAND MH and Bell Mail
  - o distribution lists

Library Routines

- o TCP (Virtual Circuit Service)
  - o device independency: system calls, stdio
  - o multiple concurrent virtual circuits
  - o byte streams & records
  - o positive acknowledgements
  - o timeouts
  - o retransmission
  - o checksums
  - o sequencing
  - o 3-way handshake
  - o windows
- o IP (Datagram Service)
  - o routing: static, dynamic
  - o multiple protocol support
  - o multiple network support
- o UNETNAMES (Host Name Mapping)
  - o functions: host name to address and back, enumerate names
  - o nicknames
- o PTTY (Pseudo-Terminal Service)

Utility Programs

- o LHOST (List HOST names)
  - o subsets: all, this, specific names
  - o long form: address, location, liaison
  - o sort by: name, number
- o TP, TU (Test Protocol, Test User Process)
  - o paths: ip loop-back, protocol to user, combinations

Vocabulary
Network
Packet
Datagram
Gateway
Internet
Host
Protocol
Virtual Circuit
Port

UFTP - UNET File Transfer Program

Mode:

Interactive	- prompt for each command and parameters
Command line	- take commands from command line

File functions:

Get	- get file from remote host
Put	- put local file onto remote host
Delete	- delete remote file
Rename	- rename remote file

Directory functions:

List	- list names of files in remote directory
Change	- change remote working directory

Automatic login

Remote !

UFTP Examples

Send a copy of the file "afile" to another host named "thathost":

```
% uftp thathost put afile
```

Retrieve a copy of the file "afile" from another host named "thathost":

```
% uftp thathost get afile
```

Print a listing of all the files in the directory "adirectory" on another host named "thathost":

```
% uftp thathost list adirectory
```

Print a listing of all the users currently logged in on another host named "thathost":

```
% uftp thathost get !who
```

UVTP - UNET Virtual Terminal Program

Complete remote terminal emulation

Escape to local shell

UVTP Example Terminal Session

```
% uvtp thathost
ThatHost UNET/UVTP Server
login: myname
password:
...now talking to shell on thathost
(escape character)!who
...continue talking to shell on thathost
(escape character)q
%
```

UMTP - UNET Mail Transfer Program

Functions:

- Send - send mail to "user at host"
- Read - read mail in my mailbox

Mail header fields:

- To - primary recipients
- From - sender
- Date - date and time message was sent
- Subject - what the message is about
- Cc - secondary recipients

Compatible with RAND MH and Bell Mail

Distribution lists

UMTP Examples

Send a message to user "joeuser" at another host named "thathost":

```
% umtp "joeuser at thathost" -subject "Meeting on 11/7/80"
Joe, I'd like to propose the following tentative agenda for
our project meeting next Friday:
        status, 10 min
        ...
        actions, 5 min
What do you think?
--Bob
(control-D)
%
```

Read mail in my mailbox:

```
% umtp
Date: 4 Nov 1980 at 2109-PDT
From: Bob at thishost
To: Joe at thathost
Subject: Meeting on 11/7/80

Joe, I'd like to propose the following tentative agenda for
our project meeting next Friday:
        status, 10 min
        ...
        actions, 5 min
What do you think?
--Bob
?
%
```

TCP - Virtual Circuit Service

Device independence:

- system calls - read, write, ioctl, etc.
- stdio - getc, putc, etc.

Basic data transfer:

- Multiple concurrent virtual circuits
- Byte streams
- Records

Reliability:

- Positive acknowledgements - end-to-end on every packet
- Timeouts - for error recovery (no negative acknowledgements)
- Retransmission - when timeout expires before ack received
- Checksums - end-to-end detect corrupted packets
- Sequencing - allows reordering of out-of-order packets
- 3-way handshake - insures reliable virtual circuit setup

Resource Control:

- Windows - limit data outstanding for flow control

TCP Examples

Open a virtual circuit to port "RPORT" on another host named "thathost":

```
fd = tcpopen(
  "thathost",           /* host name */
  RPORT,                /* remote port number */
  0,                    /* local port number = unspecified */
  0,                    /* passive flag = false */
  "r");                 /* type = read */

/* now fd can be used just like any other file descriptor */
```

IP - Datagram Service

Routing:

static - routing initialized from text file (like /etc/ttys)

dynamic - on failure a new route is chosen automatically

Multiple protocol support

exclusive - maximum of one process per protocol number

multiplexor controller - similar to multiplexed RAND PORT

Multiple network support

any medium - RS-232C, Ethernet, X.25, DMR-11, etc.

easy - each driver is a simple drop-in kernel module

IP Examples

Open file descriptor for protocol 10:

```
fd = open("/dev/UNET/proto10", 2));
```

Send an ip datagram to a remote host:

```
pwp.pwp_type = WP2ID;
pwp.pwp_uui = -1;
pwp.pwp_pps[0].pps_cnt = IPHDRLEN;
pwp.pwp_pps[0].pps_addr = &ipheader;
pwp.pwp_pps[1].pps_cnt = count;
pwp.pwp_pps[1].pps_addr = &ipdata;
pwp.pwp_pps[2].pps_cnt = 0;
write(fd, &pwp, IPHDRLEN + count);
```

Read a message from the ip device:

```
prp.prp_hdr = &uprhdr
prp.prp_tmo = 10;
prp.prp_data = &buffer;
count = read(fd, &prp, sizeof buffer);
/* on return:
        uprhdr.upr_type:        type of data
        uprhdr.upr_uui:         what user it's from (if applicable)
        uprhdr.upr_ttime:       current time
        buffer:                 the data read
        count:                  number of bytes in buffer
*/
```

Types of messages sent between ip and the protocol process:

Received from ip:

```
WU2PD           - data from user to protocol process
WP2UWA          - buffer space is available for writing to a user
WU2PIOC         - a user executed an ioctl
WTMO            - read timeout exceeded
WU2PCL          - user device closed
WI2PD           - data from ip to protocol process
WP2IWA          - buffer space is available for writing to ip
```

Sent to ip:

```
WP2UD           - data for a user
WU2PRA          - buffer space is available for reads from a user
WP2UIOC         - response to ioctl from user
WP2UCL          - response to user close
WP2ID           - data for ip
WI2PRA          - buffer space is available for reads from ip
WSETINB         - enable ip and set the number of buffers in ip q
WSETUNB         - set the number of buffers in user queues
```

Miscellaneous

UNETNAMES: Host Name Mapping Library Routines

gethnent	- get next host name entry
gethnname	- get entry given host name
gethnnum	- get entry given host address
sethnent	- rewind to beginning of host name database
endhnent	- close host name database
getmyhname	- get the host name for this system

PTTY: Pseudo-Terminal Device

master	- controlling side of a terminal
slave	- looks like a normal /dev/tty device

LHOST: List HOST names program

all	- list all names, including nicknames
this	- list this host only
specific	- list given names only
long	- list address, location, liaison
num	- sort by address rather than name

TP, TU: Test Protocol, Test User Process programs

ip loop-back	- test path from protocol to ip
p2u	- test path from protocol to user
combinations	- test user to ip to protocol path, etc.

WHAT UNET DOES

Vocabulary
Network
Packet
Datagram
Gateway
Internet
Host
Protocol
Virtual Circuit
Port

General Characteristics

- o communication software for UNIX Version 7
- o uses DoD Standard IP/TCP protocols
- o written in C
- o supports any physical network medium
- o drop-in installation

Programs

- o UFTP (UNET File Transfer Program)
  - o interactive or command line mode
  - o file functions: get, put, delete, rename
  - o directory functions: list, change
  - o automatic login
  - o remote !
- o UVTP (UNET Virtual Terminal Program)
  - o complete remote terminal emulation
  - o local !
- o UMTP (UNET Mail Transfer Program)
  - o functions: send, read
  - o mail header fields: to, from, date, subject, cc
  - o compatible with RAND MH and Bell Mail
  - o distribution lists

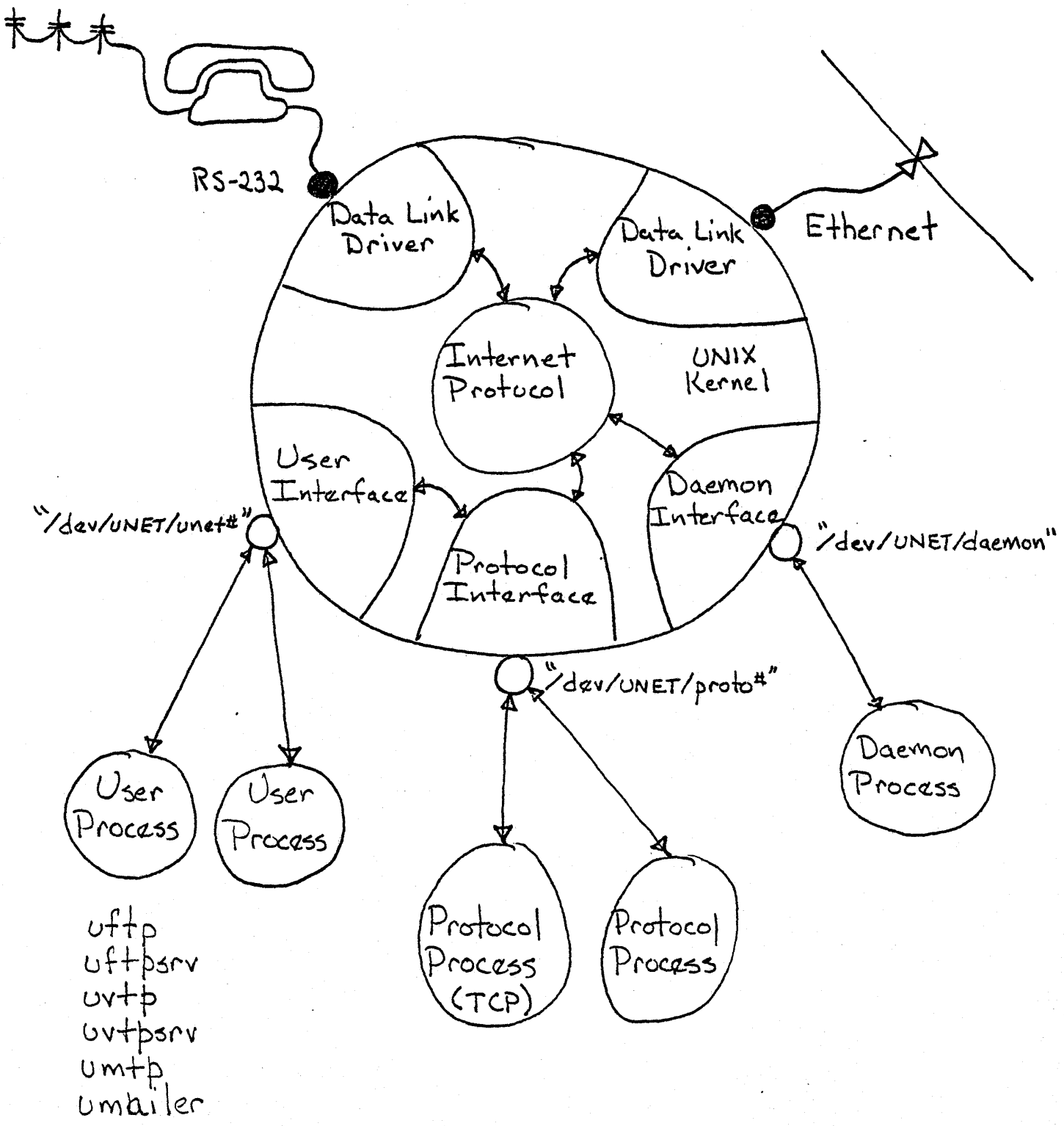
Library Routines

- o TCP (Virtual Circuit Service)
  - o device independency: system calls, stdio
  - o multiple concurrent virtual circuits
  - o byte streams & records
  - o positive acknowledgements
  - o timeouts
  - o retransmission
  - o checksums
  - o sequencing
  - o 3-way handshake
  - o windows
- o IP (Datagram Service)
  - o routing: static, dynamic
  - o multiple protocol support
  - o multiple network support
- o UNETNAMES (Host Name Mapping)
  - o functions: host name to address and back, enumerate names
  - o nicknames
- o PTTY (Pseudo-Terminal Service)

Utility Programs

- o LHOST (List HOST names)
  - o subsets: all, this, specific names
  - o long form: address, location, liaison
  - o sort by: name, number
- o TP, TU (Test Protocol, Test User Process)
  - o paths: ip loop-back, protocol to user, combinations

UNET Architecture



THREE PRIMARY INTERFACES

USER <=> PROTOCOL

PROTOCOL <=> IP

IP <=> NETWORKS

ONE SECONDARY INTERFACE

Daemon available for logging of
errors and information.

USER INTERFACE

```
                                        /* Scan for a unet device */
for ( i = 0 ; i < NUU ; i++ )
                                        /* If I find one */
    if ( open ( "/dev/UNET/uneti" ) )
                                        /* Select a protocol */
        if ( ioctl ( select protocol ) )
                                        /* Open (or listen for) a connection */
            if ( ioctl ( connect ) ) {
                                        /* Normal read/write of data */
                I / O
                                        /* Close device--get errors */
                close ( )

                exit

            } else
                                        /* Connection Failed */
                error "Can't Connect"

        else
                                        /* No Protocol Server */
            error "No Protocol Server"
                                        /* No devices available */
error "No UNET Devices Available"
```

The tcpopen and ftcpopen combine the "open" and the two ioctl calls, returning a file descriptor or standard I/O pointer respectively.

PROTOCOL INTERFACE

```
if ( open ( "/dev/UNET/proto#" ) == -1 )
        error "No Privelege"

ioctl ( set buffering )

ioctl ( enable IP )

for ( ; ; ) {

        read ( )  [ with timeout ]

        switch ( result.type ) {

        case USER_DATA

                write ( IP_DATA ) and/or
                write ( USER_READ_AVAILABLE )

        case USER_IOCTL

                [ perform action ]
                write ( USER_IOCTL_RESULT )

        case USER_WRITE_AVAILABLE

                [ update user window ]
                write ( USER_DATA )

        case IP_DATA

                write ( IP_DATA ) or
                write ( USER_DATA ) and/or
                write ( IP_READ_AVAILABLE ) and/or
                write ( USER_READ_AVAILABLE )

        case IP_WRITE_AVAILABLE

                [ update to IP window ]
                write ( IP_DATA )
                write ( USER_READ_AVAILABLE )

        case TIMEOUT

                [ find retransmission ]
                write ( IP_DATA )

        }

}
```

DATA FLOW EXAMPLE

User open, select protocol, connect, send data, close.

open("/dev/UNET/unet#")

This returns a file descriptor if the selected unet special file is not busy, and an error otherwise. No data flow.

ioctl(Select Protocol)

This ties the file descriptor to a particular protocol server--if not there, error. An ioctl data packet is sent to the protocol, including the ioctl type, and the uids, and gids (real and effective) of the requesting process. The protocol must return an ioctl result, indicating whether it will talk with this user. Success or failure is returned to the user.

ioctl(connect)

This ioctl data is sent directly to the protocol previously selected. The protocol will attempt to establish the connection, and return success or failure to the user. Connection (in the case of TCP) exchanges information with the remote host, requiring several exchanges of datagrams. The protocol then notes that the state of this user stream is connected, and which "socket".

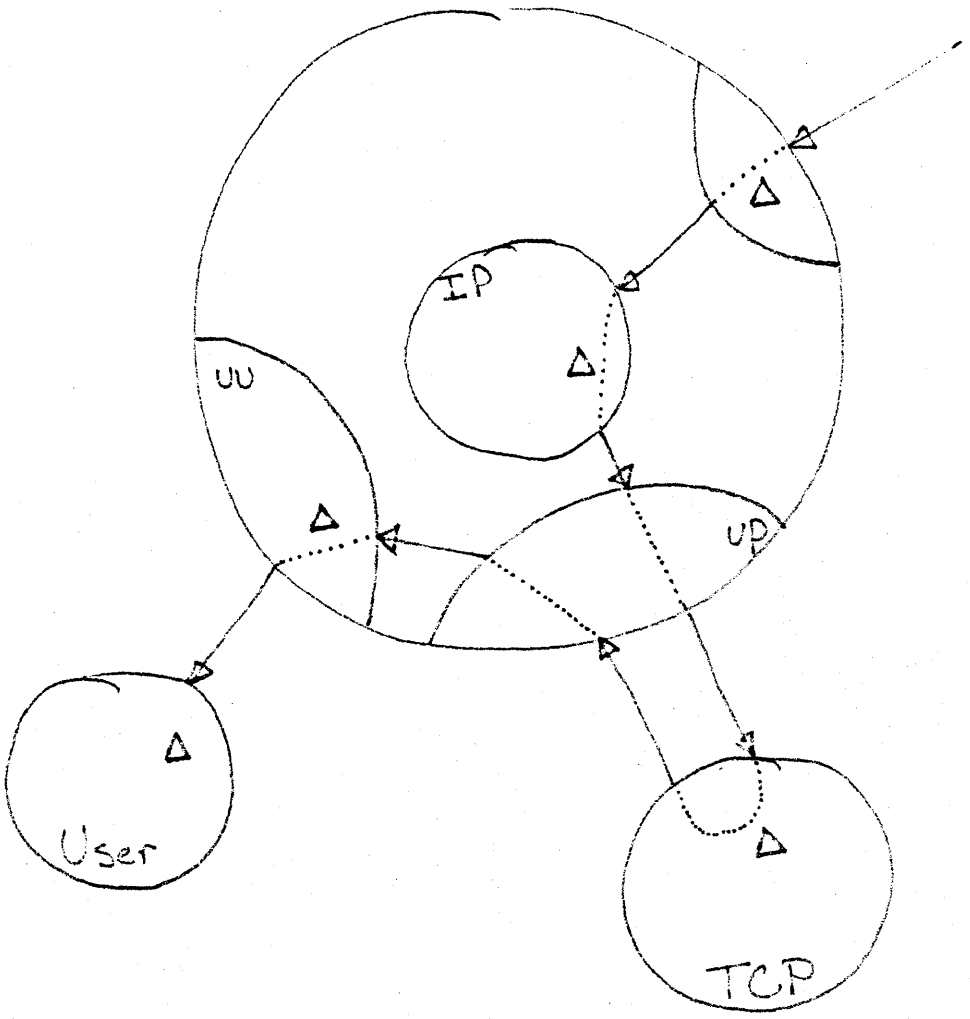
write( )

The write data is fed to the protocol as it provides read_available windows. The protocol then sends the data to the socket previously specified, queueing it for retransmission, and resending it as necessary. As the data is acknowledged, and the buffers freed, the protocol notifies the kernel that it is willing to read more data from this user connection. While there is no window to transfer data from the user to the protocol, the user will block in the write code.

close( )

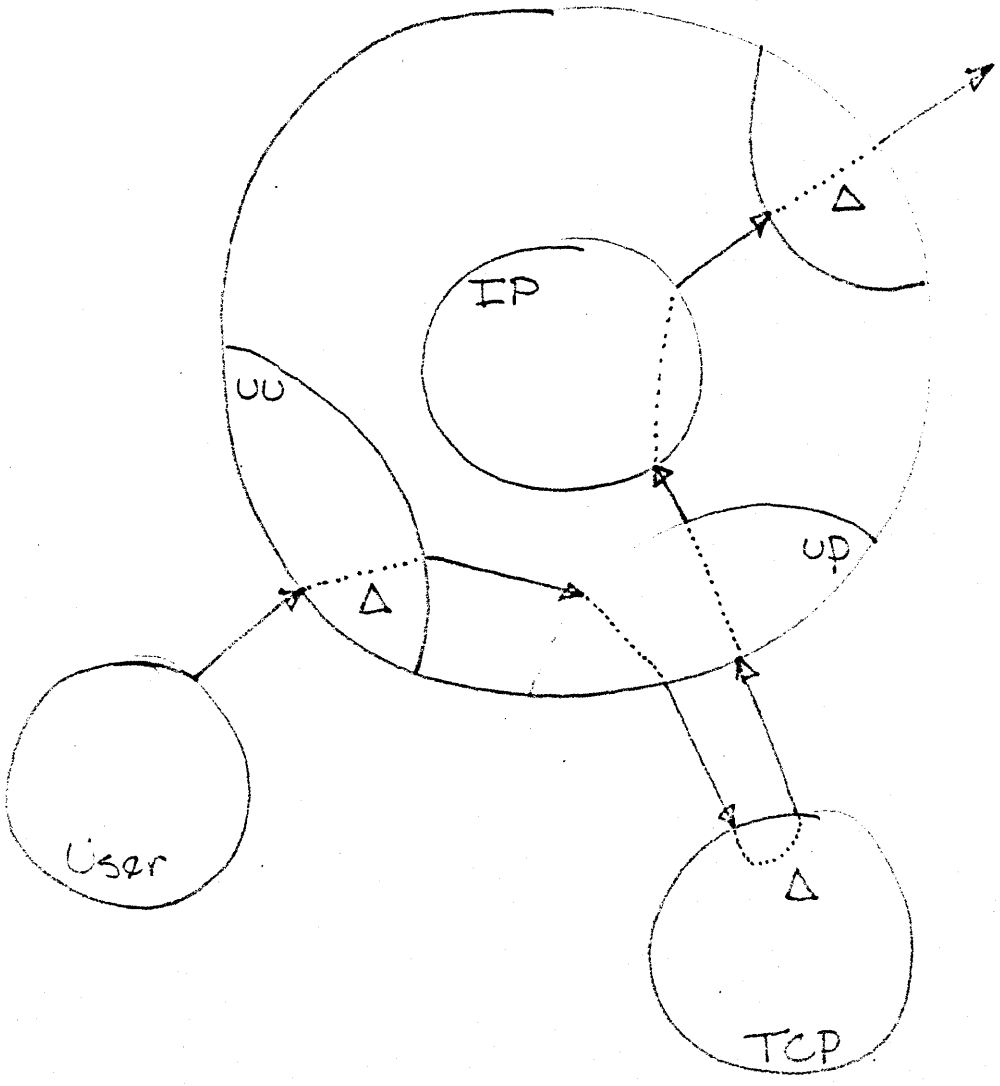
The user interface structure is flagged as CLOSED, and the user blocks. Once all of the previously written data has been fed to the protocol, a close message is given to the protocol. When the protocol has finished its close connection handshake with the remote system, it returns a close complete to the user, indicating any error status. When the user close returns, the user structure is freed for another open. If the user interrupts the close, it will return without waiting for the error result.

Receive Data



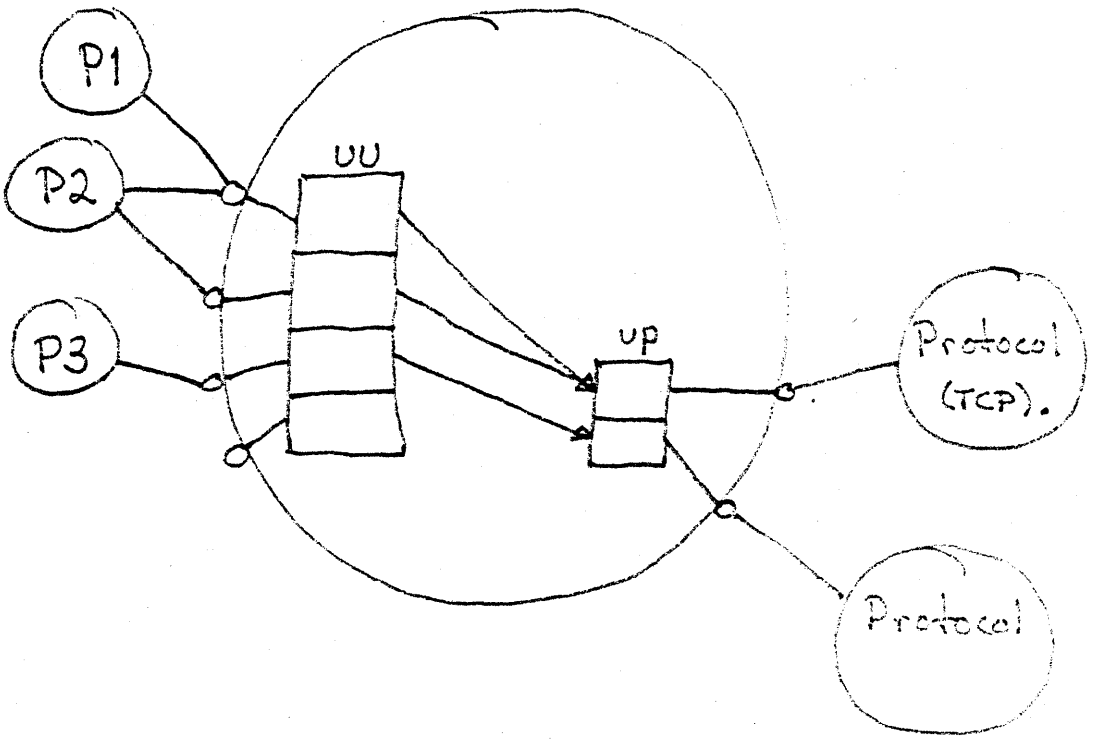
△ = buffer

Send Data



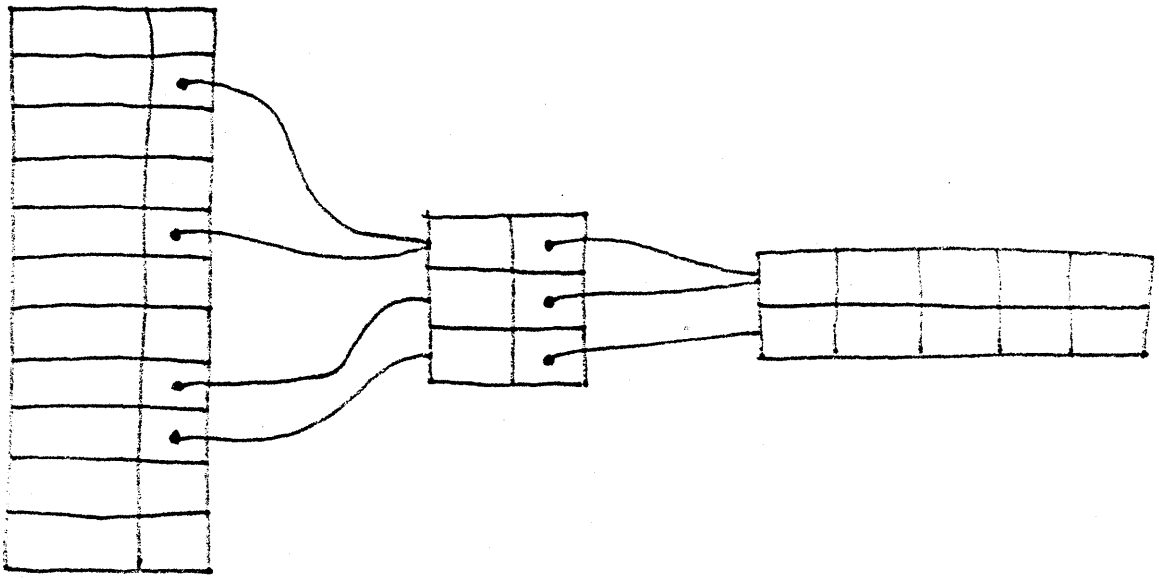
Δ = buffer

Interface Structures



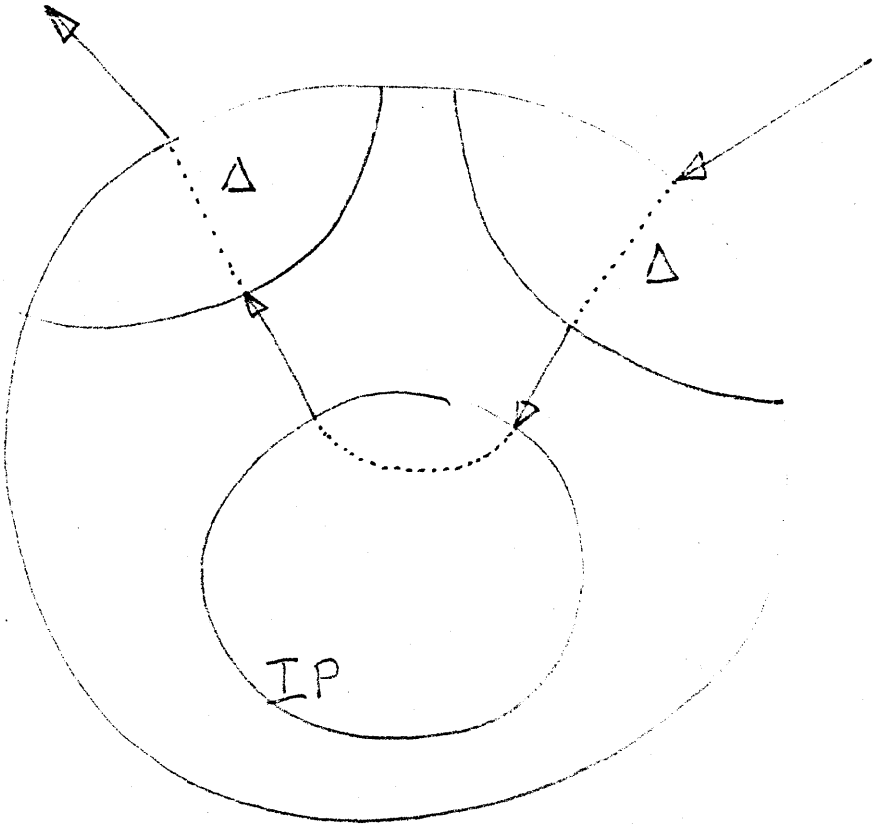
IP Routing Tables

host2dli dli2dlty dlty2dlsw



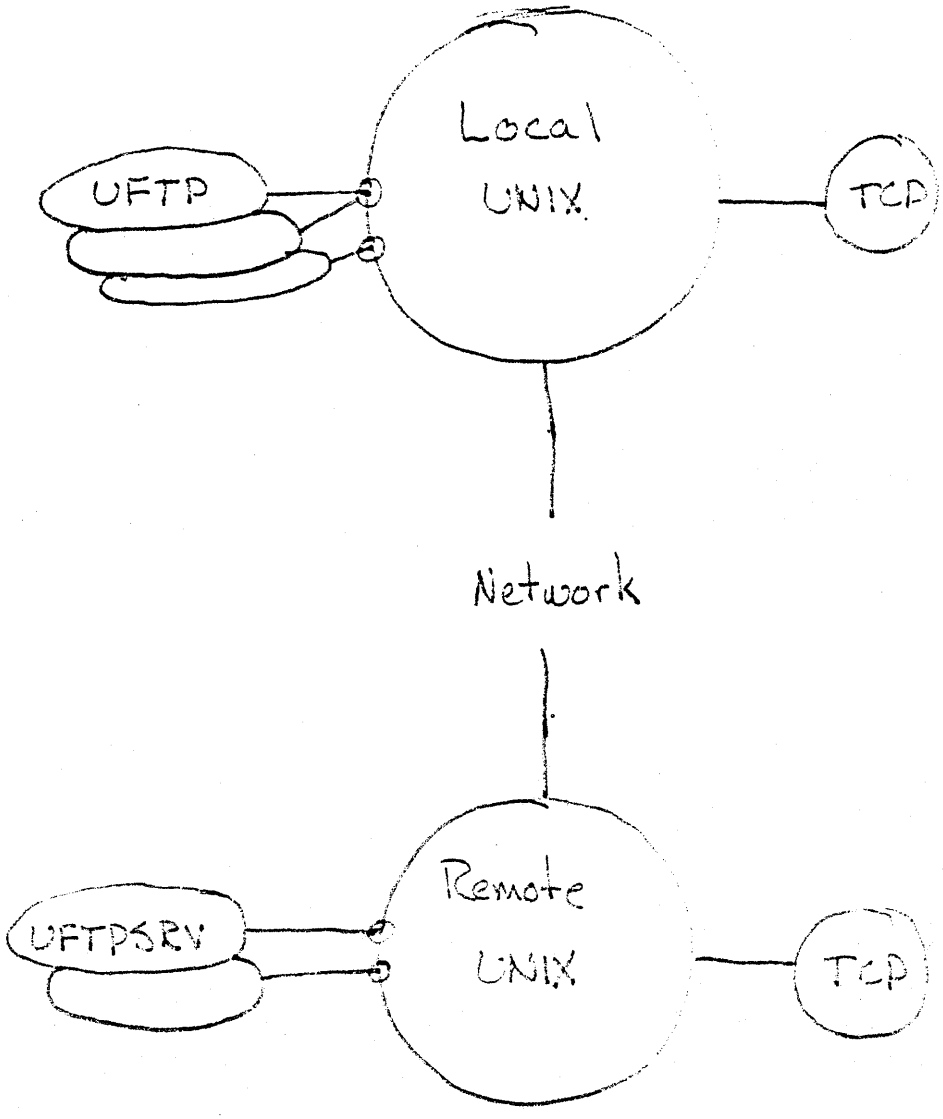
host2dli: host number to data link instance
dli2dlty: data link instance to data link type
dlty2dlsw: data link type to data link switch

IP Store and Forward

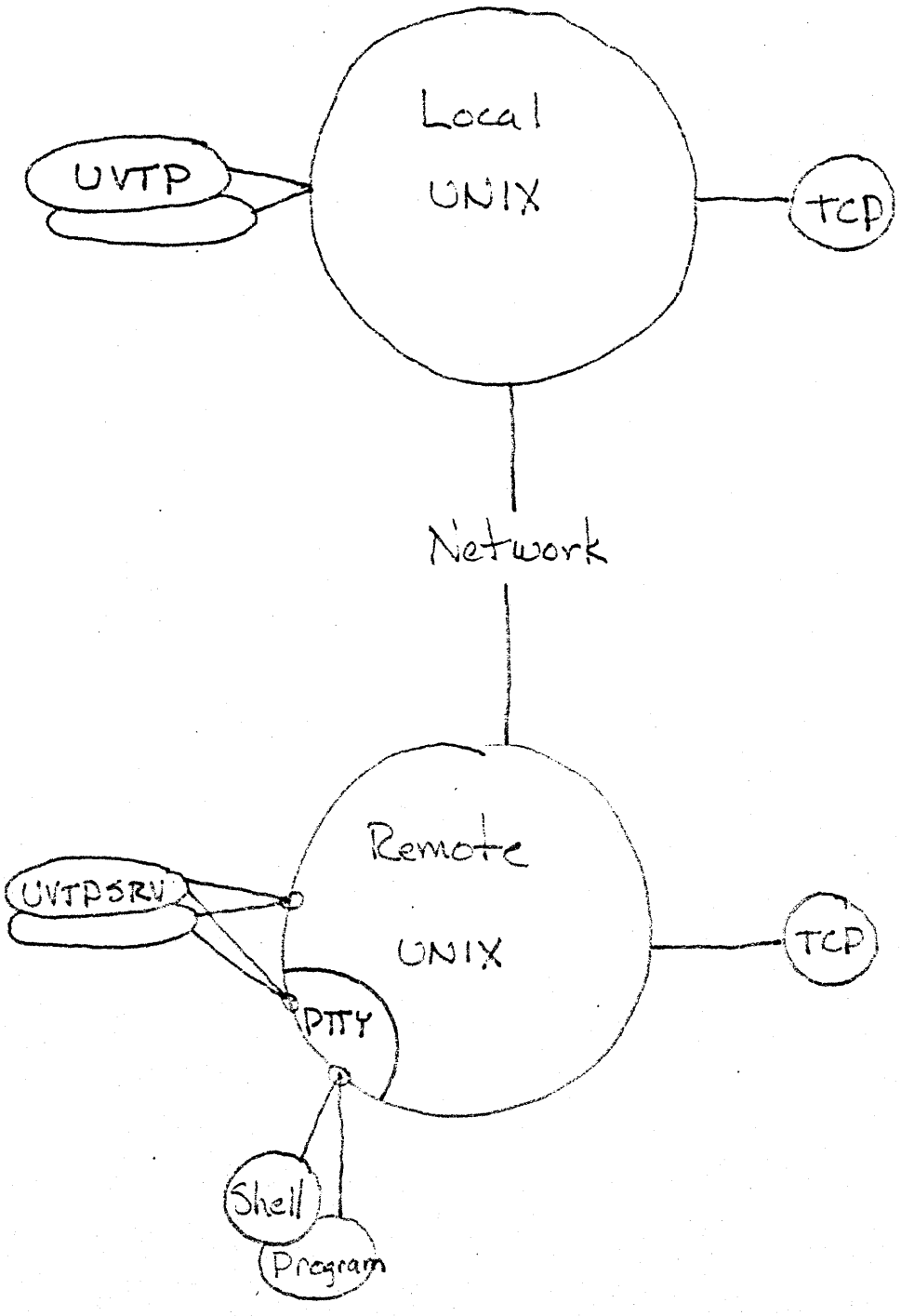


Δ = buffer

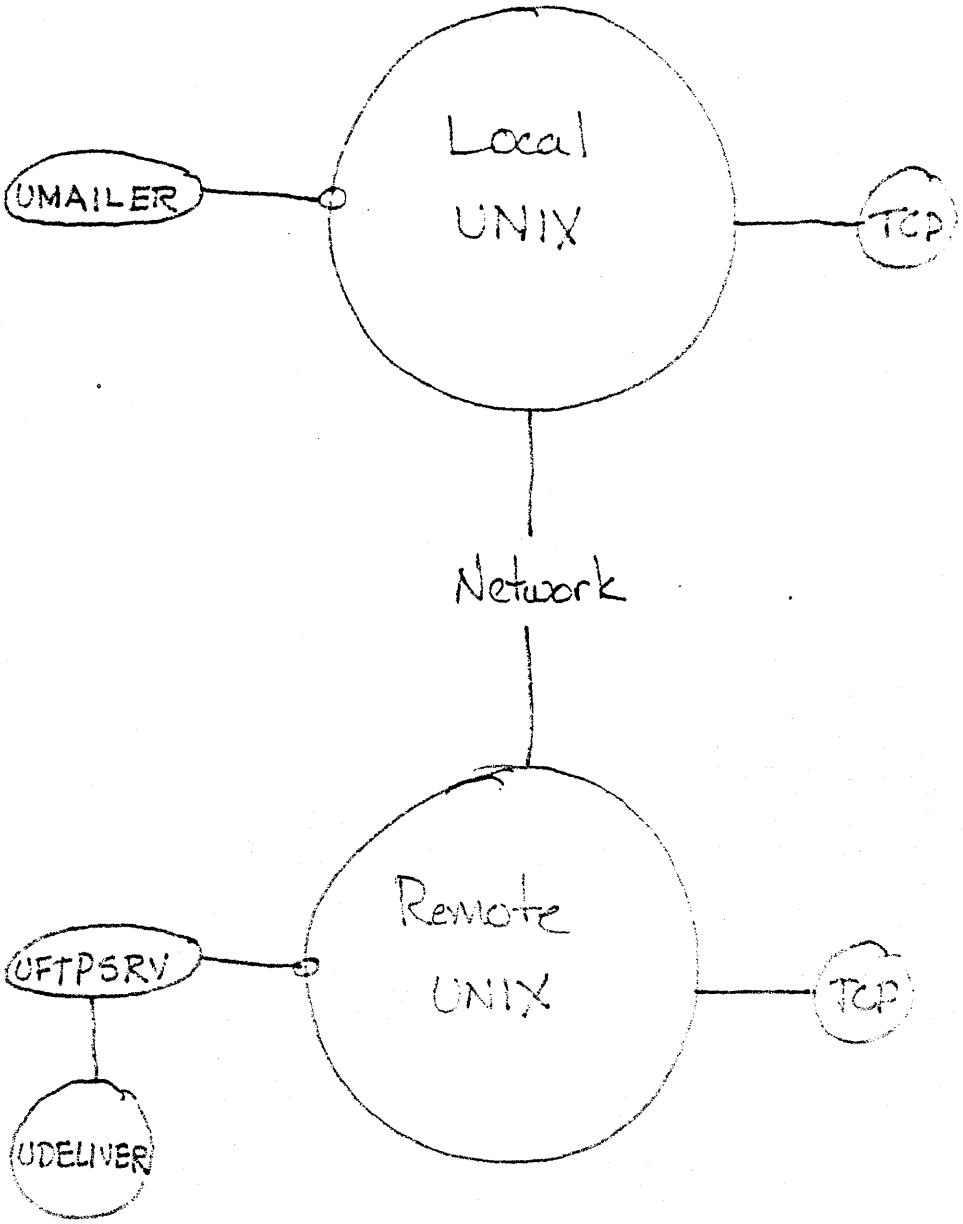
UFTP



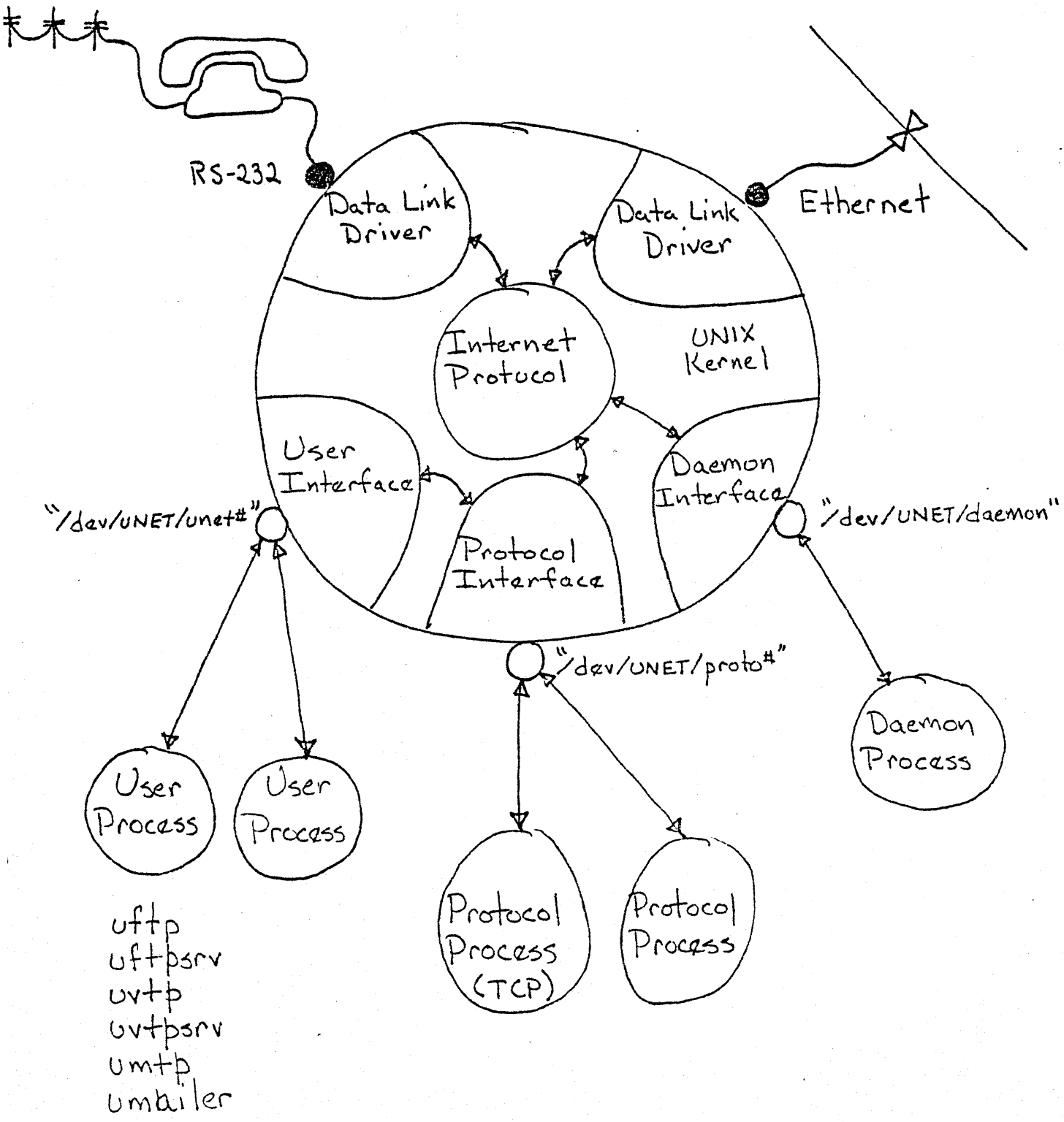
UVTP


Local
UNIX
UVTP
TCP
Network
Remote
UNIX
UVTPSRV
TCP
PTTY
Shell
Program

UMTP


Local
UNIX
UMAILER
TCP
Network
Remote
UNIX
UFTPSRV
TCP
UDELIVER

UNET Architecture



THE PROCEDURE FOR OBTAINING UNET

1. You fill out a request form and send it to 3Com at the following address:

   3Com Corporation
   3000 Sand Hill Road #1
   Menlo Park, California 94025

2. 3Com prepares license agreements requested and sends them to you (allow 2 weeks).

3. You execute license agreements and send them with payment to 3Com.

4. 3Com delivers UNET (allow 3 days for preparation of magtape and 3 days for shipment).

UNET REQUEST FORM

Yes, I am interested in obtaining UNET. I require:

[ ] Source code licenses for use on the following machines:

Quantity	CPU Model
____	_____
____	_____
____	_____

[ ] A license to distribute binaries.

Quantity	CPU Model
____	_____
____	_____
____	_____

[ ] Maintenance support.

[ ] Installation. Preferred dates: _____

_____

_____

Any special requirements? _____

_____

_____
(Name) (Title)

_____
(Company)

_____
(Building) (Mail Stop)

_____
(Street)

_____
(City) (State) (ZIP CODE)

( ) -
(Telephone)

SEND TO: 3Com Corporation
3000 Sand Hill Road #1
Menlo Park, California 94025
Attn: Bruce Borden