UUCP on NeXT™ Computers

NeXT Technical Support Notes

For Releases 1.0 and 1.0a

Written by Alan M. Marcum
Edited by Beth Dorrell
Trademark Issues by Susan Mobley
Book design by Eddie Lee

Special thanks to Andreas Sautter, Dennis Gentry, Charles Oei, Cal Thixton.

Contents

Introduction

One of the most powerful facilities on a computer is electronic data transfer — either by transferring data files, or by electronic mail. On NeXT computers, as on other UNIX® computers, UUCP is one of the facilities used for electronic data transfer over serial communication lines. UUCP, the "UNIX-to-UNIX Copy" facility, has a number of administrative idiosyncrasies — many of which we attempt to clarify in this Technical Support Note. Though UUCP is officially unsupported, we attempt a complete description of those parts of the system particularly important to correct operation of UUCP on the NeXT computer.

A brief note on terminology: UUCP refers both to an intermachine data transfer facility, and to a specific program (part of the overall facility) for copying files between computers. Whenever there is a question about which is being described, the overall facility will be called "The UUCP Facility," or "UUCP," and the program will be referenced as "the uucp program," "uucp(1)" (in keeping with the UNIX tradition of referencing a program, subroutine, or similar facility along with the chapter of the manual in which it is described), or just "uucp."

The UUCP Facility on NeXT computers is based on the version of UUCP in UNIX 4.3BSD.

This document describes the UUCP Facility as implemented in the NeXT 1.0 and 1.0a releases.

Throughout this document, "man pages" refers to the "UNIX Manual Pages," available online through the Digital Librarian™.

Road Map

If you're familiar with UUCP and need simply the NeXT-specific differences, see "NeXT versus 'Standard 4.3BSD'" on page 23.

A "Quick Setup Guide" is on page 6; an "Expanded Quick Setup Guide" begins on page 7.

Debugging hints are found beginning on page 19. Four important known bugs are described in beginning on page 22.

Sample modem configurations are listed in Appendix A, page 25.

Sample **L.sys** entries are in Appendix B, page 31.

An annotated debugging session is in Appendix C, page 34.

Suggested Reading

"Serial Port Connectivity Guide," a Technical Support Note from NeXT, Inc.

Your modem manual.

The documentation of the Zilog 8530 Serial Communications Controller, **zs**(4), in the UNIX Manual Pages, with particular attention to the cable specifications and the use of **tty*P***, **ttyd*P***, and **cu*P*** (where *P* is the port, a or b).

The various "sample" files in the **/etc/uucp/UUAIDS** directory.

Appendix F: Installation and Operation of UUCP, in the NeXT on-line system administration manual.

Three *Nutshell Series* handbooks, published by O'Reilly & Associates, Inc.:
- *Using UUCP and Usenet*
- *Managing UUCP and Usenet*
- *!%@:: A Directory of Electronic Mail Addressing and Networks*

A Brief Overview of the UUCP Facility

The principal programs of the UUCP Facility are:

Name	**Stands for**	**Program Function**
uucp	Unix-to-Unix Copy	Queues files to be copied between UNIX computers
uux	Unix-to-Unix Execute	Queues commands to be executed on a remote UNIX computer
uupoll	UUCP System Polling	Requests initiation of communications with another UUCP host
uusnap	UUCP System Snapshot	Status snapshot monitor
uuq	UUCP System Queue	Displays the UUCP Facility queue
uucico	Unix-to-Unix Copy In, Copy Out	Underlying communication program
uuxqt	uux Queue Interpreter	Underlying command execution driver

The first five of these are considered *user-level* programs, and reside in **/usr/bin**; the other two are *system-level*, residing in **/usr/lib/uucp**. During typical use as an electronic mail transfer mechanism, neither the **uucp** nor the **uux** programs are actually run by the user.

There are some additional programs associated with UUCP. They are not central to the UUCP Facility, and are mentioned here primarily for completeness. All except **uuclean** and **uucpd** are in **/usr/bin**; uuclean is in **/usr/lib/uucp**; **uucpd** is in **/usr/etc**.

Name	**Stands for**	**Program Function**
uuencode	UUCP Encode	Encodes binary files as ASCII, for transmission through UNIX mail
uudecode	UUCP Decode	Decodes files processed by uuencode
uuname	UUCP Partner Names	Lists names of known UUCP partners
uulog	UUCP Log Files	Displays UUCP log files
uusend	UUCP Send	Sends a file to a remote host
uuclean	UUCP Spool Clean-up	Removes old files in the spool directory (**/usr/spool/uucp**)
uucpd	UUCP Daemon	Handles network-based UUCP connections

UUCP has several configuration files, all in the directory **/etc/uucp**. (This is contrary to the online documentation, which incorrectly states that they are in **/usr/lib/uucp**.) The principal configuration files are:

File name	**Purpose**
L.sys	System connection information
L-devices	Description of available devices
USERFILE	Access permissions and restrictions

Additional important ancillary files, also in **/etc/uucp** are:

File name	**Purpose**
L.aliases	Hostname aliases
L.cmds	Commands which may be executed from a remote site
UUCPNAME	Alternate hostname

The UUCP spool directory is **/usr/spool/uucp**; of note in this directory are:

Name	**Purpose**
LOGFILE	Log of connections and file transfers
SYSLOG	File transfer times
AUDIT	Directory of debugging information for connection initiated remotely
LCK	Directory containing connection and device lock files

For a more information on the contents of the spool directory, see the **uucico**(8) man page.

Portions of the system which come into play when managing UUCP include:

System Component	**Role**
init, ttys, gettytab	Remote logins
tip, remote	Dial-out, and access to modems
cron, crontab, crontab.local	Periodic execution facility
sendmail, sendmail.cf	Mail delivery
UserManager, NetInfo™	Account management
rc.local	Optional UUCP Facility initialization

The **init** and **cron** programs are in **/usr/etc**; **tip** is in **/usr/bin**; **sendmail** is in **/usr/lib/sendmail**. All configuration files for these other portions of the system are in **/etc**. **UserManager** is in **/NextAdmin**.

The **uucico** program can use various (custom) communication protocols. The “g” protocol is the most widely used serial-based protocol (serial, as opposed to, for example, X.25). The g protocol is packet based, and automatically adjusts packet size based on transmission line qualities.

Quick Setup Guide

The Quick Setup Guide is designed for those who already know the UUCP Facility and UNIX, and simply want a quick overview of the steps needed to get UUCP running. For more detailed descriptions, see the Expanded Quick setup Guide (page 5) and the Detailed setup Guide (page 9).

1. Set up the modem access file, remote(5).

2. Connect the modem, per the manufacturer's instructions, *using an appropriate cable*.

3. Use tip(1) to configure the modem.

4. Set up the UUCP configuration files (in **/etc/uucp**).
   a. **L.sys**
   b. **USERFILE**
   c. **L-devices**
   d. Others, as necessary; see the detailed setup instructions, or the appropriate man pages (only needed for unusual situations).
   e. Set owner, group, permissions of configuration files.

5. Test dial-out.

6. Modify **sendmail.cf**, if desired.

7. Set up **cron**(8) entry for polling, if desired.

8. Enable and test dial-in, if appropriate.

9. If dial-in is enabled, create accounts for use by the remote sites.

10. Modify **/etc/rc.local** to execute **uucico** upon system startup, if desired.

Expanded Quick Setup Guide

For additional information, see the Detailed Setup Guide which starts on page 9.

1. Set up the modem access file, remote(5).

Make sure a line similar to

```
a2400|Dial out on cua at 2400 baud:\
      :dv=/dev/cua:br#2400:el=^C^S^Q^U^D:ie=%$:oe=^D:
```

is in the file **/etc/remote**. Replace *a2400* with the name by which you want to reference the modem through tip(1)[1]; replace *cua* by the appropriate device file name (cua if the modem is connect to Port A, cub if connected to Port B); replace the two instances of *2400* by the appropriate baud rate. See the UNIX Manual Pages, Section 5, for details.

2. Connect the modem, per the manufacturer's instructions, *using an appropriate cable*.

The correct cable is *very* important; an incorrect cable may prevent proper operation, especially with dial-in and dial-out on the same line. The correct cable configuration is noted in the **zs**(4) man page, and in the "Serial Port Connectivity Guide" Technical Support Note.

3. Use tip(1) to configure the modem.

Keys here will be the modem's responding to Data Terminal Ready (DTR), dropping (the modem must hang up the phone, or reset), and the modem's controlling Data Carrier Detect, (DCD). DCD should be high only when a remote carrier is present. On the cube under 1.0 and 1.0a, hardware flow control is not supported; if using UUCP, software flow control should be disabled.

See Appendix A for sample modem configurations.

4. Set up the UUCP configuration files.

a. L.sys

You'll probably have to use a direct (DIR) connection; the ACU (automatic call unit, or autodialer) code does not work for most modems in 1.0. Build a "chat" or "expect-send" script which controls the modem completely, assuming the modem

1 Typical convention: a or b (based on the port to which the modem is connected), followed by an indication of the speed. Another example would be **`b19.2`** (indicating a 19.2Kb modem attached to Port B).

configuration established in Step 3, above. See Appendix B for sample **L.sys** entries.

b. USERFILE

Add a line for the remote site. Comment out any lines which are inapplicable. (For example, 1.0 and 1.0a ship with access permitted for a user **Unext**, and a site **next**. These may be removed if desired.) Ensure that sufficient, but not overly permissive, access is provided.

c. L-devices

Ensure a line for a direct connection at the appropriate speed on the correct device (cua or cub) exists.

d. Others, as necessary; see the detailed setup instructions, or the appropriate man pages (only needed for unusual situations).

These files would include **L.cmds**, **L-dialcodes**, **L.aliases**, and **UUCPNAME**.

e. Set owner, group, permissions of configuration files.

All configuration files — **L-devices**, **L-dialcodes**, **L.aliases**, **L.cmds**, **L.sys**, USERFILE, UUCPNAME — must be readable by the user **uucp**. It is recommended that you have them all owned by the user **uucp**, group **daemon**, mode 600 (readable, writeable only by the owner: `-rw-------`). Note that the Edit application will always change the owner of the file to the user running Edit.

5. Test dial-out.

The uupoll(1) command will cause the remote site to be polled. Polling will occur even if no work is queued for the remote site. You may invoke **uucico**(8) directly to obtain debugging information, if necessary.

6. Modify **sendmail.cf**, if desired.

By default, the mail delivery system is configured not to initiate UUCP-based transfer of mail. Instead, the transfer is expected to be initiated by some other mechanism (such as **cron**), and so the mail is merely queued for later transfer. By removing the **-r** flag in the definition of the UUCP mailer in the **sendmail.cf** file (in **/etc/sendmail**), transfer will be initiated when the message is sent. Look for the lines:

```
Muucp, P=/usr/bin/uux, F=msDFMhuU, S=13, R=23,
       A=uux - -r $h!rmail ($u)
```

and remove the **-r** in the second line.

7. Set up **cron**(8) entry for polling, if desired.

Regardless of whether you make the sendmail modification described in Step 6, you may want to force periodic polling of the remote site. The **cron**(8) utility will facilitate this. An entry such as

```
15      7-20 * * 1-5  uucp     uupoll remotesite
```

will cause the machine *remotesite* to be polled every hour, on the quarter-hour, each week day between seven in the morning and eight in the evening. This should be added either to **crontab** or to **crontab.local**, both in **/etc**.

8. Enable and test dial-in, if appropriate.

Turn on dial-in by enabling a **getty**(8) on the appropriate port, A or B. Find a line like

```
ttyda      "/usr/etc/getty D9600"     unknown     off
```

in **/etc/ttys**. This controls the dial-in side of a serial port. Set the speed appropriately [see **gettytab**(5) for details; in general, simply change the number in the *D9600* to the maximum baud rate of the modem[2]], set the terminal type to *dialup* instead of *unknown*, and turn the line on. Inform **init**(8) of the changes [using **`kill -HUP 1`**].

Note that only one entry in ***/etc/ttys*** *should be "****on****" for each port. Do not turn on both* ***ttya*** *and* ***ttyda****, for example.*

9. If dial-in is enabled, create accounts for use by the remote sites.

Add an account for the remote machine's UUCP access to your local NetInfo domain (remember to ensure that the account's login shell is **/usr/lib/uucp/uucico**). Then, test things out by having the remote site poll your machine.

Each remote site should have a separate login name (account), though all may share the same UID. This account should be a copy of the **uucp** account, but with a different login name, real name, and password. We also recommend that a UID other than that of the uucp account be used for the remote site accounts, for improved security. (By convention, a remote site called **rhino** would use a login name of **Urhino**.)

10. Modify **/etc/rc.local** to execute **uucico** upon system startup, if desired.

UUCP is not supported in releases 1.0 and 1.0a. Furthermore, the statements in the **/etc/rc.local** shell script are commented out. It is frequently useful to have uucico run automatically upon system startup, to help ensure timely transfer of data. If you are not polling your neighboring UUCP sites, it is *especially* important that **uucico** is run when the system starts up, to guard against files staying in the queue for a long time if, for example, the system crashed because of power failure during a UUCP transfer.

[2] Note that 9600 is the fastest rate included in the default gettytab file in 1.0 and 1.0a. If you want 19200, you'll have to add an entry to gettytab. See the detailed description section, or the gettytab(5) man page.

Detailed Setup Guide

Please read the Expanded Quick Setup Guide section, if you haven't.

A bit of terminology is in order. RS-232C is a standard which specifies 25-pin pin serial communications. The nomenclature "RS-232C" is used to refer both to the signalling standard and to the connector configuration[3]. In this guide, we will be referencing three of the RS-232C signals: Data Terminal Ready (DTR), Data Carrier Detect (DCD), and Data Set Ready (DSR). DTR indicates whether the computer (the "data terminal") is ready to accept connections; DCD reflects the presence or absence of a remote carrier signal on the telephone lines; DSR indicates whether the modem (the "data set") is ready to accept commands or data.

A key idea if you're running dial-in and dial-out on the same port is that there are three different "flavors" of the port's device file. For Port A, as an example, you'll find three device special files in **/dev**: **ttya**, **ttyda**, and **cua**. All three of these reference the same physical piece of hardware. As detailed in the **zs**(4) man page, **ttya** should be used solely for hardwired devices, not for dial-up devices. When referencing the incoming side of Port A, use **ttyda**; use **cua** for the outgoing side.

The details are that **ttyda** and cua interact, and both respect the modem control lines; **ttya** does not respect the modem control lines in the same way. When an **open**(2) is attempted on **ttyda** (in other words, when a program tries to open **/dev/ttyda**), if no carrier is present as indicated by the DCD signal, the open will suspend, waiting for carrier detect. When the carrier becomes present, the open will complete. In addition, if **cua** is in use when the open is attempted, the open of **ttyda** will suspend until **cua** is closed and a carrier is detected.

If an open is performed on cua, it will cause an open attempt on ttyda to suspend. The open on cua will fail if an open on ttyda has completed.

1. Set up the modem access file, **remote**(5).

Ensure an appropriate entry exists in **/etc/remote**. See the Expanded Quick setup Guide and the UNIX Manual Pages, Section 5 [remote(5)], for details.

2. Connect the modem, per the manufacturer's instructions, *using an appropriate cable*.

Correct cables are *very* important. We know of at least four different configurations of "Macintosh®-to-Modem" cables — that is, DB-25 RS-232C to Mini-DIN8 RS-422 modem cables. Most of these configurations do not work for dial-in and dial-out access with the NeXT computer; some of them don't work even for dial-out (which is the simplest case). The difficulties lie in the connection of the modem control signal lines.

[3] The 25-pin connector referenced in the standard is also called a DB-25 connector.

To further complicate matters, some cables will work with some modems and not others. As an example, the Businessland™ Macintosh-to-Modem cable, part number 200-66688, will work fine with a Hayes® Smartmodem 9600™ V.32 modem but not with a Smartmodem 2400™. On the 9600, the DSR signal can be configured to mirror exactly the DCD signal; on the 2400, this cannot be done. This cable happens to connect both DCD and DSR to Pin 2 on the Mini-DIN (rather than just DCD, as per the specification) which is where the NeXT software expects the DCD signal. When DSR does not mirror DCD, problems can occur with this cable.

The simplest way to ensure correct cabling is to use a modem cable per the **zs**(4) man page and the "Serial Port Connectivity Guide."

3.Use tip(1) to configure the modem.

In Appendix A are sample configurations for six different modems: the Telebit® T2500, the Hayes Smartmodem 2400, the Hayes Smartmodem 9600 V.32, the Microcom® QX/V.32c, and the Fastcomm® 9696 FDX. Many other modems will work as well. The important things to note, especially with "Hayes-compatible" modems (many of which really aren't), are as follows (see also the discussion at the beginning of this section).

In order for dial-in and dial-out both to work correctly, the modem must be able to control DCD correctly. DCD should indicate the presence (or absence) of a remote carrier signal.

In order for either dial-in *or* dial-out to work correctly when the process using the modem — **uucico**, for example — terminates abnormally, the modem must be able to respond correctly to DTR dropping. A NeXT Computer will drop DTR when the device is closed. When DTR drops, the modem must at least hang up the telephone, if not perform a reset.

Failure of the modem to control and respond to these signals correctly may prevent proper operation, or may open security holes (when a connection is not terminated properly).

Hardware flow control (CTS-RTS) is not supported by a NeXT Computer's serial ports. When using UUCP, software flow control (XON-XOFF or ENQ-ACK) cannot be used, because UUCP can transmit any characters in the data stream, including those used for flow control. Configure the modem accordingly. See also the Flow Control section.

Many modems have some sort of "escape sequence" for manual transition from data transmission mode to command entry mode. There is usually a "guard time" associated with the escape sequence — some amount of time during which no characters must be received both before and after the escape sequence in order for the escape sequence to be recognized as such. If the guard time is too short, unreliable operation may result if an escape sequence happens to be transmitted in the data stream. We recommend a guard time of at least three seconds, if possible.

If your modem has "extended result codes," these can be very helpful when debugging connection problems. However, **tip**(1) may fail when these are enabled, if auto-dialing is used. We recommend that you configure the modem without extended result codes enabled if you'll be using **tip**, and that your **L.sys** chat script enables them and expects them. However, **tip** *does* expect "verbose" (English) result codes , rather than "terse" (numeric) result codes.

4. Set up the UUCP configuration files.

a. L.sys

Appendix B shows a few sample **L.sys** entries with some explanation; see also **/etc/uucp/UUAIDS/L.sys.samples**.

You may have to use a direct (DIR) connection; the ACU (automatic call unit, or autodialer) code does not work for many modems in 1.0 and 1.0a. Build a chat script which controls the modem completely, assuming the modem configuration established in Step 3, above. Setting up an ACU connection is easier than setting up a DIR connection, so you might want to try an ACU connection first if you have a Hayes-compatible modem.

b. USERFILE

The UUCP facility can be a large security hole if set up incorrectly. Proper configuration of **USERFILE** is crucial to ensuring that UUCP is *not* a security hole. Make sure that all paths in **USERFILE** are appropriate for the access granted. Be particularly wary of paths that pertain to "anyone." A sample **USERFILE** entry is

```
Urhino,rhino   /usr/spool/uucppublic
```

This permits the user **Urhino** access to the **/usr/spool/uucppublic** directory, and also provides the same access to the remote machine **rhino**.

In general, no user or machine should have access to **/***, the root directory, from UUCP*. Though there are exceptions, they are extremely rare, and they open potential security holes.

c. L-devices

The device specified in the **L.sys** file must exist in the **L-devices** file. The speed (and any modifier of the speed; see Appendix B for a brief description), the port, and the type of connection define a "device." For example, to use Port A at 19200 baud, use an entry such as

```
DIR cua unused 19200 none
```

d. Others, as necessary; see the detailed setup instructions, or the appropriate man pages (only needed for unusual situations).

You might want to add or remove commands from **L.cmds** (to allow or prevent remote execution). You may also choose to add aliases for remote sites in **L.aliases**. Recognize that for proper mail delivery, any entry in **L.aliases** must have a "minimal" entry for that site in **L.sys**. (A "minimal" entry in **L.sys** is an entry with no chat script; see Appendix B.) If you want the remote site to refer to your machine by a name other than its host name (according to the **hostname**(1) command and the **gethostname**(2) system call), put that alternate name in the **UUCPNAME** file (**UUCPNAME** is not documented in 1.0 and 1.0a).

e. Set owner, group, permissions of configuration files.

There is a great deal of sensitive information in the UUCP configuration files, including **clear text passwords** for remote sites. (A **clear text password** is a password which is not encrypted. Normally, passwords only appear in files if they are encrypted.) The files should be readable only by the **uucp** pseudouser.

If you modify these files using the **vi**(1) editor, the ownership will be maintained. If you use the Edit application, the ownership will change to that of the user modifying the files (typically, **root**). Emacs can be configured to work either way.

For proper operation of UUCP, the configuration files must be readable by the uucp user (assuming that all of the UUCP programs are SetUID **uucp**; they are shipped this way). For security, they should be readable *only* by the **uucp** user, and writeable by, at most, only the **uucp** user. For example:

```
myhost [/amm]-132% ls -lg /etc/uucp/*L*
-rw-------  1 uucp     daemon       380  Apr 17 17:06 L-devices
-rw-------  1 uucp     daemon       300  Apr 13 15:28 L-dialcodes
-rw-------  1 uucp     daemon        56  Jun  5 15:18 L.aliases
-rw-------  1 uucp     daemon       907  Jul  5  1989 L.cmds
-rw-------  1 uucp     daemon      2195  Jul 10 13:58 L.sys
-rw-------  1 uucp     daemon       414  Jul 10 13:59 USERFILE
```

Warning: A *serious* security hole will be opened if these files — especially **L.sys** — are generally readable. This affects security at your site *and* at those sites with which your machine communicates using UUCP.

5. Test dial-out.

The simplest way to test the setup is using the uupoll(1) command, and then checking the log files.

```
myhost [/amm]-137% uupoll rhino
myhost [/amm]-138% tail -f /usr/spool/uucp/LOGFILE
amm rhino (7/18-06:55-4088) SUCCEEDED (call to rhino )
amm rhino (7/18-06:55-4088) OK (startup cua 19200 baud)
amm rhino (7/18-06:55-4088) OK (conversation complete)
^C
```

(Note the Control-C to interrupt the tail command.) This shows a successful login, successful communication with **rhino**'s UUCP, and successful termination of the call.

These lines in the **LOGFILE** indicate that the initial login sequence failed:

```
amm rhino (7/18-09:05-4111) FAILED (LOGIN)
amm rhino (7/18-09:05-4111) FAILED (call to rhino )
```

These lines show that something went wrong after a successful login:

```
amm rhino (7/18-09:07-4131) SUCCEEDED (call to rhino )
amm rhino (7/18-09:09-4131) TIMEOUT (rhino)
```

The **uupoll** command causes **uucico**(8) to be run. For additional debugging information, **uucico** can be invoked directly, typically in "debug" mode. An annotated debug-mode session is shown in Appendix C. To use **uucico** directly to test the connection, use:

```
myhost [/amm]-138% /usr/lib/uucp/uucico -r1 -srhino -x4
```

Note that when **uucico** is run in debug mode, it must be run by a user who has read permission on the **L.sys** file; typically, UUCP debugging is done as the super user.

The **uucico**(8) man page provides additional details. Appendix C contains a list of the UUCP debugging messages.

6. Modify **sendmail.cf**, if desired.

Why modify the default sendmail configuration file? As shipped, when mail is sent via UUCP it is queued for future delivery — "future" meaning the next time a UUCP connection between the origin and the destination sites is made. If polling is done infrequently, there may be a significant delay encountered when transmitting the message.

These lines in the sendmail configuration file define the UUCP mailer — **uux**(1), with some arguments:

```
Muucp, P=/usr/bin/uux, F=msDFMhuU, S=13, R=23,
       A=uux - -r $h!rmail ($u)
```

One of these arguments, the **-r** flag, causes **uux** to queue the request, but not to execute **uucico** (which would cause the request files to be transferred to the remote machine). By removing the flag, the request is both queued and transferred.

7. Set up **cron**(8) entry for polling, if desired.

Why poll? If the modification in Step 6 is not made, then any queued requests will wait until the remote site contacts your machine. Even if the modification is made, there could be some difficulties. Let's examine the following scenario.

> Randy, at Safari Software's home office (hostname **rhino**), sends mail to Sandy, at Safari's Nairobi sales office (hostname **koala**). This causes rhino to place a call to **koala**. Part way through transferring Randy's mail, a solar flare causes sufficient static on the satellite telephone link that the UUCP connection is terminated abnormally. So, Randy's message is not sent.
>
> Randy doesn't send any more mail to the Nairobi office for several days; neither does anyone else on **rhino**. No one at on **koala** sends any mail to anyone at **rhino**, either. So, no UUCP connections are initiated. Meanwhile, Randy is wondering why Sandy hasn't responded.
>
> What's happened is that nothing has caused **uucico** to be run to examine the queue. If anyone at either site sends mail to the other, **uucico** will be run, and Randy's mail will be sent. Otherwise, Randy's mail will sit in the queue forever.

To circumvent this problem, we can force each machine (or either machine) to poll the other, regardless of whether work is queued or not. Recall that **uupoll**(1) causes a remote site to be polled, even if there's nothing in the queue for that site. If **uupoll** is executed automatically every so often, the problem in the above example is eliminated. The **cron**(8) daemon allows us to implement this easily.

Note also that **uupoll** can be invoked so as to queue a job and not execute **uucico**. This is useful if you want to force polling of several sites, but invoke **uucico** only once, after the last of the set of **uupoll**s is run. For example:

```
3       7-20 * * 1-5  uucp     uupoll -n remotesite1
3       7-20 * * 1-5  uucp     uupoll -n remotesite2
5       7-20 * * 1-5  uucp     /usr/lib/uucp/uucico -r1
```

8. Enable and test dial-in, if appropriate.

The file **/etc/ttys** specifies the ports on which a login is permitted. The man page [**ttys**(5)] describes the format. An associated file is **gettytab**(5), the terminal configuration database. Among other things, gettytab allows you to specify the way in which a port's speed may be changed in response to a received BREAK.

As an example of using **ttys** and **gettytab**, let's suppose we have a modem on Port B which will support either 19200 baud or 9600 baud, and we want to be able to accept calls at either speed. Put the following lines in **/etc/gettytab**:

```
# Custom dial-up rotary
A19200:nx=A9600:tc=19200-baud:
A9600:nx=A19200:tc=9600-baud:
```

This specifies that a BREAK should toggle between A19200 and A9600 (the **nx** field describes the "next" entry to be used upon receipt of a BREAK). Then, set **ttydb** in the **ttys** file:

```
ttydb      "/usr/etc/getty A19200"    dialup      on
```

The system will start up a **getty**(8) on **ttydb**, with an initial speed of 19200 baud. (Note that the **secure** flag is not set, preventing root login from a dial-in port. This is important for good security.)

You could now request the remote site's administrator to poll your machine. If the poll is done with uucico debugging turned on, the file **/usr/spool/uucp/AUDIT/***remotesite* contains the debugging output by your machine produced during the connection to *remotesite*.

9. If dial-in is enabled, create accounts for use by the remote sites.

The **UserManager** application (in **/NextAdmin**) may be used for creating these accounts. Alternatively, **NetInfoManager** may be used directly, if desired (though **NetInfoManager** is a much lower-level tool). The long view in **UserManager** should be used, so you can specify all the necessary parameters. Figure 1 shows the long view from **UserManager** for a UUCP account assigned to the site **rhino** for Safari Systems.

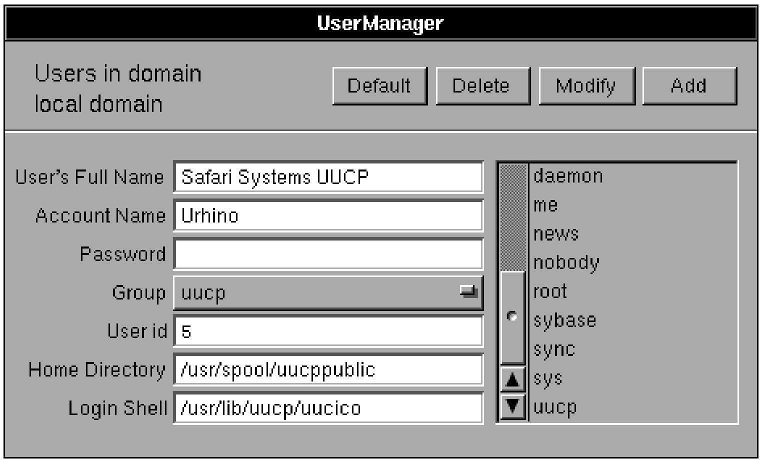

Figure 1. Long view in **UserManager**.

Note that User ID 5 was chosen. Similarly, the other remote access UUCP accounts might use the same User ID, which should be different from the User ID of the **uucp** account. (The UUCP account is, by default, UID 4; the numbering is solely by convention.)

Every account in your system should have a password. This is *particularly* important once you enable remote dial-in. Passwords for the remote access accounts should be difficult to guess, so as to minimize the chances of masquerading.

10. Modify **/etc/rc.local** to execute **uucico** upon system start-up, if desired.

After uncommenting the UUCP-related lines in **rc.local**, they'll look like this.

```
if [ -d /usr/spool/uucp ]; then
     chmod +w /usr/spool/uucp/LOG.*
     rm -f LOG.0
     if [ -f /usr/lib/uucp/uucico ]; then
          /usr/lib/uucp/uucico -r1 &
          (echo -n ' uucp')                    >/dev/console
     fi
fi
```

In addition to uncommenting the appropriate lines in **rc.local**, you might want to force an explicit polling of a remote site, particularly if that site is a central mail routing hub for you. For example, you could insert the following line just before the line invoking **uucico**:

```
/usr/bin/uupoll -n rhino
```

Add any additional local customization you feel appropriate.

Also note that there are some lines related to UUCP in **/etc/rc** which are commented out. Uncomment them if appropriate for your site.

Debugging Hints

First, make sure you have the right cable. This is very easy to do; yet, if you have the wrong cable, you might waste a great deal of time.

If you're using one of the modems referenced in Appendix A, start with the configuration from the Appendix. Only after you get the connection working should you try changing the modem configuration, unless you have compelling local needs.[4]

If you're using a modem "compatible" with a known modem, ensure that it really is compatible. This is particularly important when you're running dial-in and dial-out on the same port, when proper control of and response to the modem control lines is necessary.

If you can't complete a connection with **uucico**, try using tip(1) to do some debugging. Connect to the modem, and try to send the modem various commands. Try to connect to the remote site by typing in an equivalent of the **L.sys** chat script for the remote site. Verify the telephone number and the dialing commands (including any telephone access codes). Ensure your UUCP login and password are correct and operational.

The debug flag **-x***n* to **uucico** can be both very helpful and very overwhelming. Larger values of *n* yield more debugging output; too much of this output is useless. A rough (very rough!) guide to the **uucico** debugging levels follows.

Debug Level	Information Presented
2	Very high level connection debugging, scant little more than is in **LOGFILE**
4	Connection debugging, including most of the chat script; work queue processing
6	Complete chat script; protocol negotiation
7	Packet driver progress
11	Hostname alias expansion

The information obtained by running **uucico** in debug mode can be extremely useful in debugging connections. Each higher-numbered level includes the information produced at all lower levels. Level 4 is a good place to start; avoid levels above 7 if possible, because level 7 produces a great deal of additional output.

[4] An example of a compelling local need: the Telebit T2500 has a register, S120, to control acceptance or rejection of "micropackets." Changing this register setting can be crucial to successfully completing an overseas call.

Frequently, though not always, the **uucico** debug messages

```
pkcget: alarm 4001
pkcget: alarm 7002
```

are caused by a telephone connection of marginal quality with a high-speed modem.

Other things to check: can your machine connect to any other UUCP neighbors? Can the remote machine connect to your machine? Can the remote machine receive connections from other UUCP neighbors? Is there any useful information in the remote site's **AUDIT** file? Is there a lock file which should not be present? Does the **uucp** user have read access to the necessary configuration files?

Finally, be sure to check the section on Important Known Bugs.

Flow Control

Many people ask about flow control and UUCP. There are two sorts of flow control typically considered: software flow control, such as XON-XOFF or ENQ-ACK protocols, and hardware flow control, using the CTS and RTS modem control lines. Under releases 1.0 and 1.0a, no support for flow control is provided.

As mentioned previously, uucico cannot be used with software flow control. On the 68030-based NeXT Computer, the serial ports do not support hardware flow control.

It is, indeed, possible to overrun the CPU with a very fast modem, and thereby lose data. The packet protocol helps minimize the effects of this by providing a throttling mechanism for the data transfer rate. In addition, data checking is provided by the packet protocol to decrease the chances of undetected data loss.

We expect that hardware flow control will be supported on the 68040-based NeXTcube™ and NeXTstation™ under Release 2.0.

Important Known Bugs (Releases 1.0 and 1.0a)

There are three known bugs in the 1.0 and 1.0a releases which can significantly affect configuring and debugging UUCP connections, and one which affects UUCP operation.

The first bug: if there's a line in the **L.sys** file that is empty except for whitespace (spaces, tabs), **uucico** may crash. Why "may crash" and not "will crash"? If the almost empty line occurs in the file *after* the entry for the remote site, **uucico** will run fine; if *before* the entry for the remote site, it'll crash.

The second bug: an absolutely minimal **L.sys** entry looks like

```
remote Never none none
```

where *remote* is replaced by the remote computer's hostname. An alternative minimal entry is

```
remote Never none speed none
```

with *remote* as above and *speed* replaced by the baud rate of the connection; for example,

```
remote Never none 9600 none
```

If one uses the second form of minimal entry — to document the connection speed, for example — and the second `none` is missing, **uucico** will crash when receiving a call from that remote site.

The third bug: many of the modems which the documentation claims are supported by the Automatic Call Unit (ACU) facility of **L-devices** do not work. If you use the ACU facility and are experiencing difficulties connecting to either the local or the remote modem, try using a DIR (direct) connection instead, specifying the modem handshaking and configuration in the **L.sys** chat script.

The last bug: if the connection is over TCP rather than over a serial line, the **uucpd** at the receiving end does not run **uucico** correctly.

NeXT versus "Standard 4.3BSD"

The primary differences between "Standard 4.3BSD UUCP" and NeXT's UUCP are handling bidirectional ports and the location of the configuration files.

To use bidirectional ports, one uses a different device special file for the incoming and outgoing sides. For the incoming side of Port A, use **/dev/ttyda**; for the outgoing side, use **/dev/cua**. Both ttyda and cua reference the same physical hardware; the difference is in the way the device driver treats the presence of, absence of, or changes in the DTR and DCD modem control signals.

If you examine these files (along with those for Port B) using `ls -l`, you'll see:

```
crw-rw-rw-  1 root     wheel     11,  0 Dec 29 16:40 ttya
crw--w--w-  1 root     wheel     11, 64 Jun 22 12:01 ttyda
crw-------  2 uucp     wheel     11,192 Jun 22 12:01 cua

crw-rw-rw-  1 root     wheel     11,  1 Sep 21  1989 ttyb
crw--w--w-  1 root     wheel     11, 65 Jun  6 19:58 ttydb
crw-------  1 uucp     wheel     11,193 Jun 18 11:04 cub
```

(Dates and times in the listing are irrelevant.) Note that the minor device number — the second number in the **`11,`** *n* pair — of these files follows a pattern. The dial-in side has a minor number the same as that of the hardwired version, plus 64 (hex 40); the dial-out side, plus 192 (hex C0). These differences inform the driver to treat the devices in the special dial-out or dial-in manners.

As for the configuration files, the man pages on the UUCP facility and the on-line System Administration Manual (Appendix F) all reference the configuration files — **L.sys**, **L–devices**, and such — as residing in the directory **/usr/lib/uucp**, as they do in many systems. On NeXT computers, though, these files reside in **/etc/uucp**, *contrary* to the documentation. Similarly, the sendmail configuration files reside in **/etc/sendmail**, not in **/usr/lib/sendmail** or **/usr/lib**.

The shell script **/etc/rc.local** is run each time the system is started. As shipped, there are some statements to perform simple maintenance on the UUCP Facility, and to execute **uucico**. In 1.0 and 1.0a, these are commented out, and so are ignored. (Note the comment "UUCP is not supported in this release.") You might choose to uncomment these lines.

Ongoing UUCP Management

UUCP (and related facilities, such as sendmail) maintain a number of log files. You might want to specify the retention of additional log records as well.

All these log files consume space and should be managed (and, perhaps, analyzed or examined). The UUCP log files reside in /usr/spool/uucp, and are called **LOGFILE** (log of incoming and outgoing calls) and **SYSLOG** (log of files transferred). The sendmail facility, which handles mail delivery and dispatching, also maintains a log file, which is in **/usr/spool/mqueue/syslog** by default.

The syslog facility (distinct from the file in **/usr/spool/uucp/ SYSLOG**) can be used to record dial-up logins. Adding the following line to the syslog configuration file, **/etc/syslog.conf**, records all the "authentication" log records in the file **/usr/adm/auth.log**:

```
auth.debug    /usr/adm/auth.log
```

See the **syslogd**(8) man page for additional information.

We recommend using cron to manage and analyze these files automatically and regularly. As shipped, cron will execute **/usr/adm/daily** at 2am each day, **/usr/adm/weekly** each Saturday at 3:30am, and **/usr/adm/monthly** at 5:30am on the first day of each month. You might choose to modify those scripts, or to add your own. Sending mail to an appropriate system administrator with the results of the data analyses is a common practice.

In addition, there is a shell script **uucp.day.sh** in the **/etc/uucp** directory. This script does the following.

1. It cleans up the **C.** , **D.** and **OLD** directories of **LOGFILES** and forgotten files. (All three of these directories are in **/usr/spool/uucp**.)
2. It produces a report of the system's activity; this report is logged in the directory **/usr/spool/uucp/STATS**, and will be mailed to a master site for, perhaps, central UUCP administration. (This master site should have an alias that pipes the report through **uucp.day.sh** for archiving.) These reports can then be used to generate traffic reports. It can also generate a cost estimate based on values placed in the **L-costs** file. (**L-costs** is used only by this script; it's described by comments in **uucp.day.sh**.)
3. It queues up a set of jobs to force polling neighboring sites.

uucp.day.sh should be modified as desired, for local needs and to specify any configuration parameters (such as the "master" site in the environment variable **STATSPERSON**).

Appendix A: Sample Modem Configurations

The following sample configurations are for various modems. The presence or absence of a modem's configuration in the section is *not* an indication of endorsement, or lack of endorsement, for that modem. These are simply some popular modems with which we happen to have some experience.

Note that some of the parameters in some of the configurations are not mandatory, whereas others are. No indication of such is made in the sample configurations; consult the text of this document, the **zs**(4) man page, the *Serial Port Connectivity Guide*, and the modem's manual.

In each sample configuration, if the first line is **boldface**, it is the command to display the configuration. Anything typed by the user, including this initial command, is in **boldface**. Settings which are changed from the defaults and which are critical for correct operation are in *italics*.

All configurations support dial-in and dial-out.

Telebit T2500

```
ATn?
E1 F1 M1 Q0 P V1 W0 X0 Y0 &P0 &T4     Version GA2.00
S00=001 S01=000 S02=043 S03=013 S04=010 S05=008 S06=002 S07:070 S08=002 S09=006
S10=007 S11=070 S12:255 S18:010 S25=005 S26=000 S38=000
S41:003 S45=000 S47=004 S48=000 S49=000
S50=000 S51:254 S52:002         S54:003 S55=000 S56=017 S57=019 S58:000 S59:062
S60=000 S61:001 S62:006 S63=001 S64=000 S65=000 S66=000 S67=000 S68:000 S69=000
S90=000 S91=000 S92:001 S93:010 S94=001 S95:002 S96=001
S100=000 S101=000 S102=000 S104=000 S105=001
S110:001 S111:030 S112=001
S121=000 S130:005 S131:001
S150=000 S151=004 S152=001 S153=001 S154=000 S155=000
S160=010 S161=020 S162=002 S163=003 S164=007 S255=000
OK
```

Register numbers separated from their names by an equal sign (=) have their default values; those separated by a colon (:) have values other than the default. Registers with values other than the defaults that are important, though not critical, are S12, S92, S110, and S111.

Hayes Smartmodem 2400

```
AT&v
ACTIVE PROFILE:
B1 E1 L1 M1 Q0 V1 X0 Y0 &C1 &D3 &G0 &J0 &L0 &P0 &Q0 &R1 &S1 &X0 &Y0
S00:001 S01:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:030
S08:002 S09:006 S10:014 S12:255 S14:0AH S16:00H S18:000 S21:7CH
S22:75H S23:07H S25:010 S26:001 S27:40H

STORED PROFILE 0:
B1 E1 L1 M1 Q0 V1 X0 Y0 &C1 &D3 &G0 &J0 &L0 &P0 &Q0 &R1 &S1 &X0
S00:001 S14:0AH S18:000 S21:7CH S22:75H S23:07H S25:010 S26:001
S27:40H

STORED PROFILE 1:
B1 E1 L1 M1 Q0 V1 X0 Y0 &C1 &D3 &G0 &J0 &L0 &P0 &Q0 &R1 &S0 &X0
S00:001 S14:0AH S18:000 S21:3CH S22:75H S23:07H S25:010 S26:001
S27:40H

OK
```

Note that with both the Hayes Smartmodem 2400 and the Hayes Smartmodem 9600 V.32 some of the register settings cannot be saved in nonvolatile storage.

Hayes Smartmodem 9600 V.32

AT&V
ACTIVE PROFILE:
B1 E1 L1 M1 N1 T Q2 V1 X4 Y0 &B1 *&C1 &D3* &G0 &J0 &L0 &M0 &P0 &Q0 *&R1* &S1 &T4
&U0 &X0 &Y0
S00:001 S01:000 S02:043 S03:013 S04:010 S05:008 S06:002 S07:070 S08:002
S09:006 S10:014 S11:095 S12:255 S18:000 S25:005 S26:001 *S37:009* S59:000
S60:001 S61:000 S62:000 S63:010 S64:000 S65:000 S66:000 S67:001
STORED PROFILE 0:
B1 E1 L1 M1 N1 T Q2 V1 X4 Y0 &B1 *&C1 &D3* &G0 &J0 &L0 &M0 &P0 &Q0 *&R1* &S1 &T4
&U0 &X0
S00:001 S02:043 S07:070 S11:095 S25:005 S26:001 *S37:009* S59:000
S60:001 S61:000 S62:000 S63:010 S64:000 S65:000 S66:000 S67:001
STORED PROFILE 1:
B1 E1 L2 M1 N1 P Q0 V1 X4 Y0 &B1 &C0 &D0 &G0 &J0 &L0 &M0 &P0 &Q0 &R0 &S0 &T4
&U0 &X0
S00:000 S02:043 S07:030 S11:095 S25:005 S26:001 S37:000 S59:000
S60:001 S61:000 S62:000 S63:010 S64:000 S65:000 S66:000 S67:001

OK

Fastcomm 9696 FDX

+	bottom DIP switch:	1-8 OPEN	
ATE	Terminal Echo	ON	
ATF	Transmit data Echo	OFF	
ATM	Speaker Control	1	
ATP	Terminal Parity	Even	
ATQ	Result Code Quiet	OFF	
ATV	Result Code Verbose	ON	
ATX	Result Code Set	4	Full
	Call progress		Full
AT&B	Freeze DTE rate		ON
AT&C	Carrier Detect mode		Forced
AT&D	DTR mode	0	
AT&H	Transmit Flow Control		0 - None
AT&I	Receive Flow Control		0 - None
AT&J	Phone Jack Select	RJ-11	
AT&M	MNP and Sync mode	6	Async data , MNP ,compression
AT&N	Data rate	0	Highest Available Speed
AT&P	Dial pulse ratio	US	
AT&R	Sync RTS mode		
AT&S	DSR mode	Forced	
AT&T	Self Test Modes		
AT&Y	Break handling		
AT&Z	Stored number dialing storage		

Microcom QX/V.32c MNP Class 9 modem
SX Mode

```
+ front DIP switch:    UUDDUUUUUU
+ rear DIP switch:     DUUDUUUU


MODEM BPS       9600   SB9600
MODEM FLOW      OFF    SFM0
MODEM MODE      AUT    SMAUT (SMNOR)
AUTO ANS        ON     AAON
SERIAL BPS      9600   S1B9600
BPS ADJUST      ON     BAON
SERIAL FLOW     OFF    SF10
PASS XON/XOFF   OFF    SX0
PARITY          7E     S1P1
BREAK           5      SBRK5

 - STRIKE ANY KEY TO CONTINUE -

EXIT CHAR       000    SE0
CMD ECHO        ON     SCEON
RESULTS         ON     SCEON
RESULT TYPE     LONG   SCEON
CONN MNP-       ON     CLS1
DATA ECHO       OFF    SDEOFF
INACT TIMER     00     SIT0
AUTO RETRAIN    ON     RTON
COMPRESSION     ALL    COMP3
MAX BLK SIZE    256    BLK3
AUTO BUFF       0      AUT0
AUTO CHAR       000    SAC0
EMULATING HP    OFF    HP0

 - STRIKE ANY KEY TO CONTINUE -

PAUSE TIME      003    SPT3
DTR             2      SD2
CARR DET        0      S1C0
DSR             0      S1D0
RING IND        1      S1R1
SPKR CTRL       0      SA0
LEASE LINE      OFF    SLLOFF
DISC DELAY      000    DBD0
REM CHAR        042    SRA42
REM ENABLE      OFF    RAOFF
```

```
REM SEC         OFF    RSECOFF
ASYNC/SYNC      0      SNC0
CTS/RTS         ON     CRTS0

 - STRIKE ANY KEY TO CONTINUE -

RDLB ENABLE     ON     RLBON
DIAL MODE       ON     DCON
PULSE DIAL      US     SDP0
GUARD TONE      0      SG0
PAR CHK         OFF    PAROFF
MANUAL DIAL     OFF    SMD0
BELL            ON     BELLON
EQUALIZER       ON     EQ1
SPEED MATCH     ON     MS1
```

Appendix B: Sample L.sys Entries

All the sample entries in this appendix permit calling at any time. See the **L.sys**(5) man page for documentation on restricting call times.

1. Hayes-Compatible, 2400 Baud, Port B

   Many modems are, or claim to be, Hayes-compatible. Some are, some actually aren't. For those that are, at least to some degree, this simple **L.sys** entry might work:

   ```
   remote Any DIR 2400 cub "" ATdphone ogin: loginname ssword: password
   ```

   This is all one line and not very readable. An equivalent entry, which is much more readable, is:

   ```
   remote Any DIR 2400 cub \
         "" ATdphone \
         ogin: loginname \
         ssword: password
   ```

   Note that there's a space before the backslash in these lines. (Backslash is the line-continuation character in the **L.sys** file.)

   This entry — for *remote*, at 2400 baud, through Port B — connects to the modem, immediately sends the modem command to dial the telephone (along with the phone number), and waits for **`ogin:`** to be received from the remote machine. After receipt of **`ogin:`**, we send *loginname* (the correct login name), and wait for **`ssword:`**. Upon receipt of **`ssword:`**, we send the password.

   If the ACU code supports your particular Hayes-compatible modem, the following entry might work:

   ```
   remote Any ACU 2400 phone \
         ogin: loginname \
         ssword: password
   ```

2. Telebit T2500, 19,200 Baud, Port A

An entry such as the following works well with a Telebit T2500 modem:

```
remote Any DIR Tb19200 cua \
      ABORT BUSY \
      "" AT \
      OK~3-AT-OK~3 ATx3s50=255 \
      OK~3 ATdtphone \
      CONNECT~80 PAUSE \
      ogin:~15-CR-ogin:~10-BREAK-ogin:~5-BREAK-ogin:~5 loginname \
      ssword:~8 password
```

Besides the basics of the **L.sys** file's chat script syntax, there are some advanced features. This entry takes advantage of these facilities and of some of the modem's. We assume that the modem is in some particular initial state — specifically, that described by the T2500 configuration in Appendix A. The first line describes the remote connection. The **Tb19200** tells UUCP that a 19,200 baud modem will be used; it differs from simply using `19200` only in that it requires its own entry in the L-devices file:

```
DIR cua unused Tb19200 none
```

This is done simply for readability and maintainability; the "`Tb`" is optional (though if used in the **L.sys** entry, must be present in an L-devices entry). The second line arms the abort facility, which aborts the call if **BUSY** is received during the placing of the call.

Beginning with the third line [`"" AT \`], we begin communicating with the modem, first ensuring that we have its attention, then setting the result codes (x3 — all extended codes) and the connection speed (s50=255 — 19,200 baud). We dial the phone and wait up to 80 seconds for the message **CONNECT** from the modem. (The default timeout is 40 seconds; because of how the modem is configured, it might take up to 70 seconds for this step.)

Upon receiving **CONNECT** we wait, and then look for the string **ogin:** (eliding the initial "L" is an old UNIX tradition to allow for both "`login:`" and "`Login:`" as prompts). If `ogin:` is not seen within 15 seconds, send an empty line, and repeat. This time, though, if `ogin:` is not seen, send a BREAK (which requests the remote site to change transmission speeds; see the **getty**(8) man page for details).

If, at any point during this sequence, `ogin:` is seen, the appropriate login name is transmitted, we wait for the string **ssword:**, and, when found, send the password. At this point, the login sequence is complete, and the **uucico** programs on both sides begin their conversation.

The use of longer-than-standard timeouts is sometimes necessary; the use of shorterthan-standard timeouts may make the connection debugging process easier and quicker.

3. Minimal Entry

If your site communicates with another site, an **L.sys** entry is required. This is true even if your site never initiates the call, but only receives a call from the remote. The following is an entry which handles such a case:

```
remote Never none 19200 none
```

This entry would also be used for for aliases of polled sites (see the **L.aliases**(5) man page for details). Strictly speaking, this entry is not truly minimal; the **`19200`** could be **`none`**. But, this is a nice place to document the connection speed for possible future use.

4. Generic Format

We strongly recommend that **L.sys** entries be formatted so they are easy to read and understand (as easily as possible given a rather arcane syntax and semantics). A good generic format would be:

```
sitename connection_description \
      general_connection_issues \
      "" get_modem's_attention \
      expected_response configure_modem \
      expected_response initiate_connection \
      expected_response complete_connection_if_needed \
      expected_response wait_for_prompt \
      ogin:_with_appropriate_speed_switching loginname \
      ssword: password
```

The sample **L.sys** entry for the T2500, above, is an example of this general format. Nothing is needed for the *`complete_connection`* stage, so it's omitted. (The *`complete_connection`* portion might be used if the remote modem was connected to the remote computer by a port switch or multiplexer, for example.)

Appendix C: Annotated uucico Debug Output

`myhost#` **`/usr/lib/uucp/uucico -r1 -x6 -srhino`**	Level 6: detailed connection debugging
`root rhino (7/18-09:26-4187) DEBUG (Local Enabled)`	Time stamp, noting debugging
`finds (rhino) called`	Found system "rhino"
`getto: call no. cua for sys rhino`	No phone number found for rhino
`Using DIR to call`	Using a direct connection
`Opening /dev/cua`	Using **/dev/cua**
`login called`	Beginning login sequence from chat script
`ABORT ON: BUSY`	Abort is enabled (abort the call if " `BUSY` " is received)
`wanted """"`	Wanted nothing
`got: that`	Received what we wanted
`send "AT"`	Sending the `AT` — attention — command
`wanted "OK~3"`	We'll wait three seconds for "`OK`"
`AT`	Modem echoes "`AT`"
`OKgot: that`	Modem sends "`OK`"; that's what we wanted
`send "ATX3s50=255"`	Send more modem commands
`wanted "OK~3"`	...and wait 3 seconds for acknowledgment
`>ATX3s50=255`	The > is the modem's prompt
`OKgot: that`	
`send "ATdt9w1-800-555-1212"`	Dial the phone
`wanted "CONNECT~80"`	The modem responds "`CONNECT`" after the connection
`>ATdt9w1-800-555-1212`	
`CONNECTgot: that`	
`send "PAUSE"`	We got "`CONNECT`"; wait a bit (might get a prompt)
`wanted "ogin:~15"`	
` FAST/UUCP/COMP`	Part of the modem's "`CONNECT`" message, from the x3
`NeXT Mach (rhino) (ttyda)`	Prompt sequence begins. If it hadn't, we might have sent a newline (CR), or a BREAK.
`rhino login:got: that`	Got a prompt ("`ogin:`"); we'd have waited 15 seconds
`send "Umyhost"`	Send things to log in (user name, password)
`wanted "ssword:~10"`	
` Umyhost`	
`Password:got: that`	
`send "a5c#.9jq@Z"`	

`root rhino (7/18-09:27-4187) SUCCEEDED (call to rhino )`	The chat script has completed
`imsg looking for SYNC<`	We're looking for a "greeting" from the remote UUCP
`\20>`	(this is all before the data transmission protocol starts)
`imsg input<Shere=rhino\0>got 11 characters`	This is what we received
`omsg <Smyhost -Q0 -x6>`	Sending this back, identifying ourselves, at debug level 6
`imsg looking for SYNC<`	Looking for an acknowledgment of receipt
`\20>`	
`imsg input<ROK\0>got 3 characters`	Got the acknowledgment
`msg-ROK`	
`Rmtname rhino, Role MASTER,  Ifn - 5, Loginuser - root`	Initial connection complete
`rmesg - 'P' imsg looking for SYNC<\20>`	Begin negotiating data transfer protocol
`imsg input<Ptfg\0>got 4 characters`	Remote sends "Ptfg"
`got Ptfg`	Remote knows these protocols
`wmesg 'U' g`	We want the "g" protocol (dial-up serial line)
`omsg <Ug>`	
`Proto started g`	Starting the g protocol
`protocol g`	
`root rhino (7/18-09:27-4187) OK (startup cua 19200 baud)`	Protocol startup message
`*** TOP ***  -  role=MASTER`	We're starting up, in master role
`wmesg 'H'`	Nothing to send; does the remote have anything?
`rmesg - 'H' got HY`	
`PROCESS: msg - HY`	
`HUP:`	
`wmesg 'H' Y`	
`cntrl - 0`	Nope, nothing on the remote; shut down
`root rhino (7/18-09:27-4187) OK (conversation complete)`	Termination message
`send OO 0,omsg <OOOOOO>`	Cleanly shutting down
`imsg looking for SYNC<\20>`	
`imsg input<     \10*"   \20>`	
`imsg input<     \10*"   \20>`	
`imsg input<     "*\10   \20>`	
`imsg input<     "*\10   \20>`	
`imsg input<OOOOOO\0>got 6 characters`	Connection shutdown complete

Document Revision Notes

This document is revised from time to time. The following lists the release date of each version of this document, and the changes from the previous version.

Date	**Changes from last version**
July 18, 1990	First preliminary version created, for 1.0 and 1.0a
September 2, 1990	Second preliminary version
October 1, 1990	First released version