MAKING RS-232-C COMMUNICATIONS PACKAGES WORK WITH A DIGITAL PBX

Christine Gianone and Frank da Cruz
Columbia University Center for Computing Activities
New York, NY 10025

15 May 1989

Organizations are people. What happens when a large organization embarks on a massive conversion of its data communications infrastructure? The ultimate effect is on the people sitting at their personal computers, running the software packages they are accustomed to, accessing the applications and services required by their work. Communications software packages represent an enormous investment in both time and money. Can this investment be preserved when an analog telephone voice network and a separately switched terminal network are replaced by a single digital voice/data private branch exchange? In this article, we will address a very specific concern: how RS-232-C communications software packages (see *Shopping for software that lets PCs with mainframes*, December 1987) can be adapted to the new environment.

Several years ago, during the early days of disvestiture, Columbia University began to investigate of the possibility of replacing its AT&T Centrex telephone equipment with a university-owned private branch exchange (PBX). Ownership of the switch and wiring plant has its advantages, primarily cost containment and operational control. The high capital expenditure could be offset over time by the expected savings in operating costs. Columbia's request for proposals included a requirement for simultaneous data and voice capabilities, as well as for extra wire pairs at each wall jack to accommodate future data expansion.

The university is now in the throes of installing an IBM 9750 Business Communications System, which is composed of an IBM 9751 Computerized Branch Exchange (CBX) Model 70 connected to desktop ROLMphones (see *Inside IBM's new PBX, or the search for an office networking hub*, January 1988). When the project is complete, 75 buildings on and around Manhattan's Morningside Heights campus will be completely rewired, with the wiring plant extending to virtually every office, classroom, and dormitory room. For off-campus buildings, a total of a dozen cable trenches will be dug across the local streets. In all, 6,000 miles of cable will be laid. About 50,000 wire pairs will be connected to the main distribution frame in Columbia's central administration building, Low Library. One hundred thousand pairs of riser cables will be installed, and a similar number of station cables will be connected to about 13,000 voice/data jacks, about 3,500 of them in dormitories. Whereas data installations were once difficult or impossible, often requiring months of lead time, the new wiring plant will make them quick and simple.

The cutover will affect all those who use the university's computers because the 9750 Business Communications System will replace not only the Centrex phone equipment, but also the Gandalf PACX IV data switch, which provides access to Columbia's central mainframe computers. Because of fundamental differences between Centrex and the new switch, the installation will also affect those who dial out from their offices to external services.

When the new switch is fully installed, it will provide intra-switch connections, dialin connections, and dialout connections. Each of these presents its own conversion problems, and choosing among complicated options may be critical to achieving satisfactory data communications.

Although the present discussion is specific to the IBM 9750 Business Communications System, it also applies in large part to the Rolm and IBM/Rolm CBX products, such as the CBX II 9000 and the VSCBX 8000. The problems addressed would likely surface during installations of digital PBXs from other vendors as well. For the remainder of this article, the IBM 9750 will be referred to as "the CBX."

Data Calls to Campus Computers

For the past 12 years, terminals at Columbia University have been connected to Columbia's central mainframes through a Gandalf PACX IV data switch. PACX connections come in the form of an LDS-125, a limited-distance modem on the user's desk with four wires carrying analog signals back to the PACX. To connect to a particular computer, the user turns on the LDS-125, types a carriage return for speed recognition, then gets a menu of the available systems, and is prompted to select one:

```
Welcome to PACX IV SE.

Valid node names are: HOSTA, HOSTB, HOSTC, MYHOST

Enter node name => 
```
(user types system name here)

For example, the user might type MYHOST. The PACX maintains a table of computer host names and associated ports. When a known host is selected, the PACX finds the first free port and establishes a circuit between it and the user's terminal port, as shown in Figure 1. When dialing in from off-campus, the LDS-125 is replaced by the user's modem, and the user dials the number of a modem hunt group connected to the PACX. Once the dial connection is complete, the procedure is the same.

Under the new regime, Centrex phones will be replaced with special units manufactured by IBM/Rolm, called "ROLMphones." These can be voice-only, or voice/data. The data option adds a 25-pin RS-232-C connector and accompanying electronics to the single-line or multiple-line voice telephone. Through a single pair of wires, a data-equipped phone can manage a voice conversation simultaneous with an asynchronous data connection up to 19.2 Kbit/s, using a proprietary protocol known as "ROLMlink." This single instrument replaces the Centrex telephone, the LDS-125, and the user's dialout modem.

When the PACX is replaced by the new CBX, the dialog will change. As before, the user types a carriage return for speed recognition, but the prompt is different, and the user must type "call" in front of the hostname.

```
                  Gandalf PACX                         ___________
                                                      O           |
                  _______________________             O  Host A   |
terminal ----- O                        O             O           |
terminal ----- O                        O             O___________|
terminal ----- O                        O              ___________
terminal ----- O                        O             O           |
   :                                    O             O  Host B   |
   :                                    O             O           |
 modem ------- O                        O             O___________|
 modem ------- O                        O              ___________
 modem ------- O                        O             O           |
 modem ------- O________________________O             O  Host C   |
                                                      O           |
                                                      O___________|

(** REPLACE WITH ARTWORK **)
```

Figure 1: The Gandalf PACX Data PBX

```
CALL, DISPLAY OR MODIFY?
call myhost
CALLING 74258
CALL COMPLETE
```

In this case, the user types "CALL MYHOST". This superficial change in format should not pose an insurmountable problem. In all likelihood, users have been interacting with the PACX "manually" all these years, and conversion to the new switch will be a minor psychological adjustment. Dialing in from off-campus also remains the same as before except for the change in syntax.

The Modem Problem

Matters are more complicated for those who want to dial off-campus sites from campus phones. To understand why, it helps to know something about modems and their relationship to the CBX.

A modem, or "modulator-demodulator," converts the digital signals of a computer's RS-232-C serial communications port to analog waveforms suitable for transmission on voice-grade telephone lines, and vice versa (Fig. 2).

Modems come in two major varieties: acoustically coupled and directly connected (Fig. 3). Acoustic coupling is susceptible to noise and vibration; therefore, it only works under good conditions, and only up to speeds of 300 bit/s or 1.2 Kbit/s.

Whether internal or external to the computer, direct-connect modems reduce potential problems and increase the maximum speed by providing a direct electrical connection to the telephone wall jack. Furthermore, these modems have access to all the wires in the phone

```
   RS-232-C        Analog phone line        RS-232-C
PC------------modem---------------------modem-----------host
  (digital        (wavy analog waves)
   square
   waves)
```

Figure 2: A Traditional Modem Connection

A picture of an acoustic coupler, an external direct-connect modem, and an internal modem

Figure 3: Acoustic vs. Direct-Connect Modems

circuit, so they can simulate all the functions of a telephone: detecting and answering a ring, going off-hook, and even dialing a phone number.

ROLMphones cannot be used with direct-connect modems, however, because the proprietary ROLMlink signaling that takes place between the telephone and the CBX is incompatible with the signaling methods used by modems (Fig. 4). Thus, it serves no purpose to plug a modem into the ROLMphone's wall jack -- the modem and the switch will not be able to communicate.

Hundreds of people at the university have direct-connect modems in their offices or dorm rooms. If they cannot use these modems with their new ROLMphones, then how can they call off-campus computers, outside data services, and computer bulletin boards?

```
                                                    Analog
   RS-232-C           ROLMlink      RS-232-C        phone line
PC------------ROLMphone-----------CBX-----------modem---------------

(phone line, cont'd)       RS-232-C          (** SEE ARTWORK **)
---------------------modem----------host
```

Figure 4: Dialing Out from the IBM Switch

IBM's solution to the dialout problem is to provide the switch with a central outbound modem pool, in which modems are assigned dynamically to those who request them. The pool has many advantages: it is shared by all of the univerity's computer users, so economies of scale are achieved; it is centrally maintained; and it can be upgraded as new technologies appear. It may be used by anyone having a ROLMphone with the data option, plus a terminal or personal computer with a serial communications port and a modem cable. (Those who have an internal modem only, with no serial port, will have to get a serial port and cable.)

However, the modem pool has one key disadvantage: the conventions used for call placement are not the ones used by the Hayes modem. This seems a trivial point until we begin to look at the popular communications software packages that are actually in use on campus. Most of these packages expect to be running on personal computers that are connected to Hayes or compatible modems and, therefore, "speak" to the modem using the Hayes AT command set. The switch, however, does not accept AT commands, and most communications software packages are not written with the IBM 9751 CBX call-placement dialog in mind. How can these software packages be made to work with the new CBX?

Alternatives

To accommodate software packages that require the Hayes command language for dialing, IBM offers the 244PC. This is a special four-line data-equipped ROLMphone which contains a built-in Hayes AT command set interpreter. But because this is a multiline telephone, the expense to the user is greater than for the normal single-line phone with data.

It is also possible to furnish users with analog, rather than digital, telephone sets. But here too, the cost is higher than a for a digital line, in this case because extra interface cards are required in the CBX. Analog phones could be used with Hayes or other direct-connect modems, either internal or external. Furthermore they may be necessary for devices that use modems, but which use nonstandard data transmission techniques on the digital end, such as Telecommunication Devices for the Deaf (TDD's) that use a 5-bit Baudot code. And analog phones are also required for use with facsimile machines and other special-purpose devices that have built-in modems.

As third alternative, it was discovered that the ROLMphone can be used at low speeds with acoustically coupled modems, or acoustic adapters for direct-connect modems. But these devices cannot be used for autodialing because the CBX does not respond to the pulse or DTMF dialing tones which are generated by autodial modems, in their mimicry of real telephones. Nevertheless, the growth of the digital PBX market could result in a mini-boom for acoustic couplers and adapters, catering to those who would rather live with low speeds and without autodialing than pay the monthly data service surcharge: one step forward, two steps back...

In a large organization like Columbia University, a great deal of money can be saved by avoiding the 244PC and the analog phone wherever possible. This can be done if the popular communication software packages can be adapted to the CBX's normal call setup procedure.

Dialout Mechanisms

Why should there be such a fuss over what the user must type to place a data call? Let's look at the question in more detail.

The dialing language of an autodial modem allows users to place calls by typing simple commands from a terminal or personal computer. But first, the cables must be correctly connected and the modem turned on (blunders along these lines can happen to the best of us). Then the communications software must be set for the desired transmission speed and parity, and the personal computer or terminal must be sending the Data Terminal Ready (DTR) signal to the modem. Then the software can be connected to the modem, and the user can type dialing commands.

The Hayes AT command language is simple but powerful. A typical dialog with a Hayes modem (or a 244PC, which emulates the Hayes modem) goes something like this:

`ATZ F1 Q0 V1 X1 S0=0`	*(User types modem initialization string)*
`OK`	*(Modem responds "OK")*
`ATDT 9,7654321`	*(User types dialing command)*
`CONNECT 1200`	*(Normal modem response)*

The user's commands begin with `AT`, the modem's command introducer. The initialization string puts the modem in a known state, so that the software can control it properly. In the sample shown above, the `Z` resets the modem to default settings; `F1` puts it in full-duplex no-echo mode; `Q0` instructs it to produce result codes; `V1` specifies that the result codes should be words (for example, CONNECT or BUSY) rather than digits; `X1` selects the extended result code set (for example, CONNECT 1200); and `S0=0` disables auto-answer. Once initialized, the modem replies "OK", and the user types the dialing command ATDT followed by the phone number. When the other modem answers, the user sees the message "CONNECT 1200".

Most software packages expect to be controlling a Hayes modem. The user simply supplies the telephone number; the software supplies the appropriate AT commands to the modem and interprets the modem's responses (OK, CONNECT, BUSY, NO ANSWER, etc.), all of which is invisible to the user. Packages like Smartcom (Hayes Microcomputer Products, Inc.) and Crosstalk (DCA/Crosstalk Communications) are able to present the user with a dialing directory, from which the user chooses the number to be dialed. The package does the rest.

On the IBM 9751 CBX, call placement dialog is through Interactive Call Setup (ICS), which is markedly different from the Hayes dialog:

1. The user selects the desired communications speed and parity (for example, 1.2 Kbit/s, even parity) and ensures that the personal computer is asserting the DTR signal.
2. The user transmits a carriage return so the CBX can determine the transmission speed.
3. ICS, after some delay, responds with "CALL, DISPLAY OR MODIFY?". The user should not transmit until this prompt has appeared.
4. The user transmits "call 976543210" (or any desired number), followed by

carriage return (the first two digits represent a request for an external line).

5. There is a delay while the CBX attempts to make the connection. Various messages may appear, such as "CALLING 74250", which should be ignored. Characters transmitted to the CBX during this period will be lost.

6. If the call was placed to a local host, and it was completed, the message "CALL COMPLETE" is issued, and the user is connected with the host. If the user was dialing out through the modem pool, the connection is made silently, with no call complete indication. If the connection could not be made, the message "CONNECTION FAILED" is issued.

7. Characters may now be transmitted to the host computer. In most cases a carriage return is required to get the host's attention.

Making the Software Connection

In general, communication software packages can provide three ways to establish a connection to a remote computer:

1. **Direct**, meaning there is no dialing procedure. The user just "connects" and starts interacting with the remote computer immediately. This method is generally provided for use with PCs that are hardwired to computers with no intervening modems, but it allows the user to get at the dialing language of a modem (or data switch) that the package doesn't support.

2. **Dialup**, usually configured for Hayes and/or other types of modems, where the user can configure the package to dial new kinds of modems (or data switches) not explicitly supported by the program. Since no known software package has built-in dial support for ICS, the dial feature can only be used with ICS if the user can tailor the dialing mechansism.

   The dial feature is important because it is often associated with a phone directory, in which each entry contains not only the phone number, but also the associated communications settings.

3. A **script**, in which the user writes a program in the package's script language to interact with a remote computer or other device either directly or through a modem. The script generates what the user would type, and reacts to responses the same way the user would.

   Scripts are important for two reasons. The first is convenience, since they allow routine, repetitive interactions to be done by the computer rather than the human. "Pushbutton" applications can be set up for administrative or data entry personnel to minimize training and headaches. The second reason, perhaps more important, is that scripts allow unattended operations. One computer can call up another when everyone is asleep and phone rates are low to transfer files or exchange mail. Networks like USENET and FIDO depend on this kind of unattended operation.

The communications software packages used at Columbia were checked for their ability to handle each of these methods. Several packages are discussed in detail but many more were evaluated (Table 1). If the package allows direct connection, then it may be used with the CBX. If the package includes a dial feature, then this was tested for adaptability to the CBX. Finally, if the package has a script language, scripts were written to place calls from the CBX through ICS. Note that scripts and certain dial commands depend on receiving a call completion indication from the "modem". Because the IBM 9751 CBX provides a CALL COMPLETE message when calling a local host, but not when dialing out through the modem pool, different methods must be used in each case.

MS-DOS KERMIT 2.31

Kermit is a file transfer protocol developed at Columbia University, so packages based on it are popular on campus. All Kermit programs support direct (no-modem) connections and thus work with the CBX. Many commercial communications packages incorporate the Kermit protocol, but this is no guarantee that they can be adapted to the CBX (although most of them can). The MS-DOS version of Kermit for the IBM PC and compatibles is used not only for file transfer but also for VT102 and Tektronix terminal emulation in hundreds of locations throughout the university.

Like all Kermit implementations, MS-DOS Kermit is easy to install and run. No lengthy configuration dialog is required. The user simply starts the program, sets the minimum parameters necessary for communications (usually just speed and parity), and then uses the CONNECT command to begin terminal emulation. This allows immediate communication with whatever is connected to the PC's serial port: the PACX, a modem, or a ROLMphone (Fig. 5).

(What the user types is underlined; a terminating carriage return is implied).

```
A>kermit
IBM-PC Kermit-MS Version 2.31
Type ? or HELP for help

Kermit-MS>set speed 2400
Kermit-MS>connect
```

(User types a carriage return here)

```
CALL, DISPLAY OR MODIFY?
call 976543210
```

(User waits for a beep from the phone, then types carriage return.)

```
login:
```

Figure 5: MS-DOS Kermit Direct Connection to CBX

In addition, MS-DOS Kermit allows scripts to be written to automate any procedure that the user would do manually. For example, the script shown in Figure 6 could be used to call the number 6543210 through the CBX and begin the login process on the remote computer. This script may be created using any text editor and saved in a file. The Kermit program can be instructed to execute the script file by using the TAKE command, specifying the file's name. The file can contain any valid Kermit commands, but the INPUT, OUTPUT, and PAUSE commands are special for scripts because they simulate

```
set input timeout quit    ; Quit if desired input doesn't appear
set speed 1200            ; Set desired speed
output \13                ; Carriage return (ASCII 13) wakes up ICS
input 10 MODIFY?          ; Wait up to 10 seconds for ICS prompt
pause 1                   ; Wait one second before replying
output call 976543210\13  ; Call the phone number (97 dials outside)
pause 20                  ; Wait 20 seconds for completion
output \13                ; Send a carriage return to the host
input 20 login:           ; Wait for "login:" prompt
connect                   ; Let the user take over
```

Figure 6: MS-DOS Kermit Script for Dialing Out from the CBX

what the user would do during terminal emulation. OUTPUT sends the characters that the user would type, and INPUT reads the characters that are received from the other computer, looking for the specified character sequence. The command SET INPUT TIMEOUT QUIT tells Kermit that if the requested input does not appear within the specified time interval, then the script should not be continued. These commands do what the user would do, so they perform a kind of "human emulation".

When a call is made to a local computer attached as a host to the CBX, a CALL COMPLETE message is issued. The annoying twenty-second pause required in the dialup case is eliminated so that the script can proceed immediately upon receipt of the message (Fig. 7). A single script could be used in both cases, provided Kermit was instructed to

```
output call myhost\13     ; Call the computer named is "myhost"
input 20 CALL COMPLETE    ; Wait up to 20 seconds for completion
output \13                ; Send a carriage return to the host
input 20 login:           ; Wait for "login:" prompt
connect                   ; Let the user take over
```

Figure 7: Taking Advantage of CALL COMPLETE

allow for immediate response when the CALL COMPLETE message appears, and a mandatory pause when it does not (Fig. 8). In this case, SET INPUT TIMEOUT QUIT is replaced with SET INPUT TIMEOUT PROCEED to tell Kermit not to give up if the expected output did not appear.

Kermit scripts illustrate the general characteristics of a script language: the ability to send characters to a remote device, to receive characters and compare them with a given pattern, to set time limits on selected operations, to pause for specified intervals, and to quit when the script is obviously not working. Scripts also include some kind of notation for special characters that could not be included otherwise, such as Kermit's `\13` for carriage return (13 is the ASCII value for that character).

This has been a simple example of a script, but it's not very useful as it stands because it

```
set input timeout proceed ; Ignore timeouts, keep going.
input 20 CALL COMPLETE    ; Wait up to 20 seconds for completion
output \13                ; Send a carriage return to the host
set input timeout quit    ; From now on, quit script upon timeout
input 20 login:           ; Wait for "login:" prompt
connect                   ; Let the user take over
```

Figure 8: Allowing for Local and Dialout Connections

can be used only to dial a particular phone number. It would be much more convenient to be able to define a command, "dial," that would perform this procedure for any telephone number, and some script languages allow this. For example, Figure 9 shows how this could be done in MS-DOS Kermit. Here the new command "dial" is defined to execute all the

```
define dial set inp tim q,set sp 1200,out \13,in 10 MODIFY?,-
  pau 1,out call \%1\13,pau 20,out \13,in 20 login:,connect
```

Figure 9: Defining a New Kermit Command

commands from the previous example (note how Kermit allows commands to be abbreviated), but with the desired phone number substituted for the formal parameter "`\%1`" at the time the command is executed, a la MS-DOS batch. With this definition in place, the Kermit user could type "dial 975551212" or "dial 34321" (a local extension), or any other number. For even greater brevity, the user could define shorthand commands to dial frequently used numbers:

```
define d1 dial 975551212
define d2 dial 54321
```

Kermit, like many other packages, will automatically execute any commands found in a special "initialization file" upon startup. This is a good place to keep definitions such as these, so that they will always be in effect every time you run the program.

Scripts can also have features for decision making: For example, the Kermit script shown in Figure 10 will work with either a Hayes modem or the IBM 9751 CBX. This script works by issuing a Hayes command and looking for the OK response. If the OK response does not appear, then the script issues an ICS command instead. For Hayes modems, the script dials the number, and then looks for the Hayes CONNECT response. The full message should be CONNECT 1200 or CONNECT 2400. Even though the dialup speed is 2.4 Kbit/s, the modem on the other end may have answered at 1200, in which case the Hayes will automatically fall back to that speed, and inform the software in its CONNECT message. The "reinput" commands examine the previous material to see if it contained a "1200" or "2400", allowing Kermit to change its speed accordingly. With some extra work, this script could be reworked into a dial command similar to the one shown previously, so that it could be used for dialing any desired number.

```
; A Kermit script that tries different dialing methods.
; First, define a command to handle dialing failure:
define giveup echo Dialing failed, hangup, stop
; Now the script itself...
set speed 2400                    ; Set desired speed.
:HAYES                            ; Let's see if it is a Hayes modem.
output ATZ F1 Q0 V1 X1 S0=0\13    ; Initialize modem at this speed.
input 10 OK                       ; Look for Hayes response.
if failure goto cbx               ; Not found, go try CBX dialing.
echo Hayes dialing...             ; It's Hayes, tell the user.
output ATDT 97,555-1212\13        ; Dial the number and <CR>.
input 30 CONNECT                  ; Look for modem's confirmation.
if failure giveup                 ; Give up after waiting 30 seconds.
reinput 3 2400                    ; Was the message CONNECT 2400?
if success goto good              ; If so, we're ready to go.
reinput 2 1200                    ; No, how about 1200?
if failure giveup                 ; Neither, so quit.
set speed 1200                    ; It was 1200, so adjust speed.
goto good                         ; Go issue message and connect.
:CBX                              ; Not a Hayes, so try CBX dialog.
reinput 10 MODIFY?                ; Look for CBX prompt.
if failure giveup                 ; If not found, give up.
echo CBX dialing...               ; It's the CBX, tell the user.
output call 975551212\13          ; Dial the number.
pause 20                          ; Wait for call to complete.
:GOOD                             ; We got through, somehow.
echo Connected!                   ; Display a message.
connect                           ; Start terminal emulation.
hangup                            ; When done, drop DTR to hang up.
```

Figure 10: A Kermit Script That Works With Both Hayes and CBX Dialing

Finally, a script can contain looping constructs. For instance, if we wanted Kermit to try dialing three times before giving up, we could write a script program containing the SET COUNT and IF COUNT statements (Fig. 11).

The basic principles of manual connection and of script construction illustrated by Kermit apply also to many other popular communications packages, though the form and and range of options available can vary dramatically.

PROCOMM 2.2

ProComm 2.2 is a popular "shareware" communications package from DataStorm Technologies (Columbia, MD). Like Kermit, it does not require a lengthy configuration or installation procedure. After starting the program, you may begin terminal communications immediately, assuming the speed and parity are set appropriately. The command Alt-F10 displays a help screen, and Alt-P allows the alteration of communications parameters and with the option to save them in a file so that ProComm starts up with the desired settings thereafter.

ProComm is perfectly suitable for "manual" use with the IBM switch. It is also possible to

```
set count 3                     ; Loop limit.
goto first                      ; Skip pause on first try.
:AGAIN                          ; Label for top of loop.
echo Trying again...            ; Tell the user what's happening.
pause 30                        ; Pause for 30 secs between calls.
:FIRST                          ; Label for first try.
dial 977654321                  ; Try dialing.
if success goto online          ; If it worked, let user take over.
if count goto again             ; Otherwise try again.
echo Try again later.           ; If too many tries, give message,
stop                            ; and stop.
:ONLINE                         ; Get here on successful connection.
echo Connected!                 ; Let user know.
connect                         ; Begin terminal emulation.
hangup                          ; Hang up when the user returns.
```

Figure 11: A Script That Keeps Trying

configure ProComm for automatic dialing through the switch, using either its built-in dialer support or its script language.

To set up automatic dialing, type Alt-S, select (1) MODEM SETUP, and then set the following:

1. Modem initialization string: Just make it null (empty).
2. Dialing command: "`!~~~~~~~~~~call `" (note the blank after "`call`"). The exclamation "`!`" is ProComm's notation for a carriage return (CR), and the tilde "~" is used by ProComm to represent a 1/2 second pause. This command sends the required CR, pauses 5 seconds, and then gives the CALL command.
3. Connect string: "`CALL COMPLETE`" (only used for redialing)

For dialing out from the CBX, omit the connect string to disable redialing since there would be no connect string in that case. This highlights the most basic difficulty in adapting software to the new environment. The switch does not inform the software in any way -- not by message, not by modem signal transition -- when the connection has been made. This fact, combined with the switch's built-in queuing for busy resources, renders useless the redial feature found in so many communications programs.

Having defined a dialing command for the CBX, now type Alt-D and add the relevant phone number to the dialing directory. Then to dial a host, type Alt-D and select the number of its directory entry. Once these commands are debugged, the configuration can be saved by answering Yes to the question in the menu.

Dialing is not reliable, however -- it just forges ahead without checking that valid responses are coming from the CBX. If the CBX is slow in responding to the initial CR, the CALL command will be issued prematurely and ignored by the CBX. Figure 12 shows a ProComm script that can be used to synchronize the call.

The script file can be created with an editor and saved with a filetype of `.CMD`, and executed

`%E1`	*(Emulate a VT100)*
`%F"3"`	*(2400 bps, even parity)*
`%U2`	*(display carriage return as-is)*
`%U4`	*(full duplex)*
`%T"^M"`	*(send a carriage return)*
`%I"CALL, DISPLAY OR MODIFY?"`	*(wait for prompt)*
`%T"call 976543210^M"`	*(issue CALL command, followed by CR)*
`%T"~~~~~~~~~~~~~~~~~~~~~~^M"`	*(pause, then send carriage return)*
`%I"login:"`	*(wait for the host's "login:" prompt)*

Figure 12: A ProComm Script for Use with the CBX

using Alt-F5. Unlike a Kermit script file, the ProComm script may not include comments.

CROSSTALK XVI VERSION 3.61

Crosstalk XVI is a popular commercial communication package for the IBM PC family. It is designed primarily to be used with modems, but it also allows direct connections.

To use Crosstalk in direct mode to dial out through the CBX, type "xtalk" to start the program, and then, after the logo screen disappears and the status screen appears, type one or two carriage returns until you get the "Command?" prompt on the bottom line. At this point, use any of the two-letter commands listed on the top part of the screen to set the necessary communications parameters. For example, when connecting to a DEC VAX, type:

`SP 1200`	*(Speed 1200 bps)*
`PA E`	*(Parity even)*
`DA 7`	*(Seven data bits)*
`DU F`	*(Duplex Full)*
`EM VT`	*(Emulate a VT100 terminal)*

Then establish the connection as follows:

`GO LOCAL`	*(Make the connection)*
	(Type a carriage return)
`CALL, DISPLAY OR MODIFY?`	*(Get CBX prompt)*
`call 976543210`	*(Type the call command)*
	(Type a carriage return)
`login:`	*(Here's the VAX's login prompt)*

You can get back to Crosstalk by typing its attention character, normally Escape, and you can redefine the attention character with the AT command. Once you have got this procedure working, you can use the SA (save) command to save it in a file (for example, MYHOST, which is automatically created with the filetype `.XTK`) and it will appear in the startup menu, or you can run it at any time using the Crosstalk LOAD command.

Crosstalk also lets you set up command files for dialing different computers. A Crosstalk command file is most easily created by editing a previously saved `.XTK` file. The trick is to replace the Hayes AT dialing commands with the switch's equivalents. Relevant commands are:

```
Aprefix                            (Remove the "Answer prefix")
NUmber    976543210                ("number" to dial for MYHOST)
DPrefix   |~~~~~~~~~~call          (dial prefix: send CR, pause, then "call ")
DSuffix   |                        (dial suffix: carriage return)
GO                                 (dial the number)
```

The vertical bar "`|`" is Crosstalk's symbol for carriage return, and each tilde means to pause for one second. Note that the dial prefix must have a space after the word "call". A Crosstalk command file is similar to a ProComm dial entry and has the same drawback -- if the hard-coded pause interval is not long enough, then the "call" command will be sent before the CBX is ready for it. To circumvent this difficulty, you can write a Crosstalk script to synchronize and verify the call setup procedure (Fig. 13). The easiest way to make

```
go local                          (connect directly to ICS)
wait delay 10                     (wait one second)
reply |                           (send a carriage return)
wait string "MODIFY?"             (wait for prompt)
reply call 976543210|             (dial the number)
wait delay 200                    (pause 20 seconds)
reply |                           (send a carriage return to UNIX)
wait string "login:"              (wait for UNIX "login:" prompt)
```

Figure 13: A Crosstalk XVI Script

a Crosstalk script file is to copy a command file to another file with filetype `.XTS` and then use an editor to replace the GO command with script commands. A script may be executed at any time using the Crosstalk DO command.

SMARTCOM II VERSION 2.2

Smartcom is a Hayes product for the IBM PC family designed for use with Hayes modems. Fortunately, it also allows for direct connections, which are the "back door" into the CBX. You cannot do automatic dialing of the CBX with Smartcom (except through a 244PC), because the only dialing devices it supports are those made by Hayes.

Smartcom II searches for a Hayes modem connected to one of the ports, but since your PC is now connected to a ROLMphone, the program eventually gives up with a message "Smartmodem not responding". At that point, you can get into the main menu by pressing the F1 key. Configuration is a rather involved affair: choose "2" from the main menu (Edit Set), then choose one of the predefined Setups (Set), then F1 back to the "Edit Set" menu, then choose "Parameters", and then specify that the connection type is DIRECT, and modify the speed, parity, and other communications parameters as required. Then select "begin communication". At this point, you may type carriage return to get the "CALL, DISPLAY OR MODIFY?" prompt, and type the desired "call" command.

The ICS dialog may be automated by the use of a "macro." To define a macro you must visit the menus again: from the main menu select 2 (Edit Set), then from the Edit Set menu choose S (which Set to edit), and then choose the unused set X. Then select M to define a macro and then Z to assign it the label Z (automatic logon). At this point Smartcom II presents you with a table, which you fill in (Fig. 14). The columns read across, Wait this

```
MACRO DEFINITION                                      Press F2 For Help

Name Of Macro: Z - Automatic Logon                       Set X - Unused
Time-out      Prompt       Data                                 Send CR
    0          0 (off)                                            YES
   20         63 ("?")     call myhost                            YES
   20         69 ("E")                                            NO
    2         69 ("E")                                            YES
   10         58 (":")                                            NO
```

Figure 14: A SmartCom II Macro

long (Time-out) to get this character (Prompt) from the remote system, and then if the desired character arrives or if the time limit expires, send this character string (Data), followed by a carriage return if (Send CR) is YES. In the example, the first line waits no time for nothing, then sends nothing followed by a carriage return. Then Smartcom waits up to 20 seconds for the question mark from the CBX's `CALL, DISPLAY OR MODIFY?` prompt. Prompts can be single characters only, and are indicated by the decimal ASCII value. When the prompt arrives (or the timer goes off) Smartcom sends the text "call myhost" followed by a carriage return.

In this example, we are showing a call to a local host rather than a dialout call, to illustrate an item of interest. We want to wait for the "CALL COMPLETE" message before sending a carriage return to the host to get its "login:" prompt. But Smartcom II macros only let us look for single characters, not words or phrases. And the phrase "CALL COMPLETE" has two E's in it. So the script waits up to 20 seconds for the first E (allowing time for call placement) and sends nothing. Then it waits 2 seconds for E number two, responds with a carriage return, then waits 10 more seconds for the colon "`:`" from the "`login:`" prompt.

For dialing out, the switch does not give us the CALL COMPLETE indication, so the last three lines of the table would be omitted, and "myhost" would be replaced by a telephone number.

Once the table is filled in correctly and tested, it can be saved by typing ESC and then R (Record); it will appear on the Set X macro menu the next time you run Smartcom II.

Smartcom II is not the only package that restricts input matching to single characters. VTERM III, from Coefficient Systems (New York City), works the same way. In that package, scripts are assigned to function keys, which the user has to press, e.g.

```
Shift F1: <CR><W:?><P:1>call 976543210<CR><P:10>
```

Which means send a CR, Wait for a question mark (the one in "MODIFY?"), pause 1 second, send the call command and a carriage return, and then Pause for 10 seconds.

SMARTCOM III VERSION 1.0A

Like Smartcom II, Smartcom III can be used with either Hayes modems or direct connections. Before you can communicate, you have to go through a configuration process that is not entirely intuitive. First, you must define an "Activity" *and* a "Connection." Each has a menu and several submenus that must be traversed. For each, choose New to create a new entry, and give it a name, go through the menus and make the appropriate choices, and then press F8 to save it.

An Activity contains information that applies to the host or service you're connecting to: what kind of terminal to emulate, what the terminal settings should be, which file transfer protocol to use, how to set up your keyboard, etc. For example, you might create an Activity called VAX, in which you emulate a VT-102 terminal in full duplex, and use the Kermit protocol for file transfer. A Connection contains the data communication parameters: the communication port, speed, parity, data bits, etc. The key item here is Type of Connection: you must specify Direct. For example, you might create a Connection called CBX, which specifies Direct, COM1, 2400 bps, even parity, 7 data bits.

Having created an Activity and a Connection, you can return to the main menu and select Initiate a Session. Smartcom will give you menus of all the available Activities and Connections, and you must choose one from each. Having done all of this (it won't take more than half an hour!), you may now begin to communicate with the switch. Type carriage return in the normal way, get the CALL, DISPLAY OR MODIFY? prompt, and you're on your way. In the future, you can take advantage of the configuration work you've done by invoking the program with the Activity and Connection name on the command line, as in

```
C>scom3 vax.cbx
```

Like Smartcom II, Smartcom III includes a large dialing directory, listing major services like Compuserve, Dow-Jones, The Source, MCI Mail, etc. Presumably, access to these services is one of the reasons you bought Smartcom. How can you continue to dial them from your new CBX phone?

Smartcom III includes a script language as well as a remarkable feature called "learned scripts". When you turn on script learning (from the session menu), it will observe both what you typed in and the responses that come back. It then translates the observed interactions into a program in Smartcom III's script language, SCOPE. Figure 15 shows an example for logging into MYHOST through the CBX. Note that Smartcom automatically discarded the lines (like "CALLING 74250") that did not evoke user typein, and it timed the responses. The same script should work for dialing (substituting the phone number for "myhost".) If the CALL COMPLETE message fails to appear within 20 seconds, the script will go on to the next statement.

How can this script be combined with our predefined Activities? Just give the script the same name as the Connection that you use to get to that Activity, in our case "CBX". Whenever you establish a "CBX" Connection, the script executes automatically, and dialing up your favorite service with Smartcom III is as painless with the CBX as it was with a

```
{ Smartcom III  Learned Script   7/19/88  7:28 PM }
TYPE KEY RETURN_KEY ;
WAIT FOR PROMPT "CALL, DISPLAY OR MODIFY?" , 10 ;
TYPE LINE "call myhost" ;
WAIT FOR PROMPT "CALL COMPLETE" , 20 ;
TYPE KEY RETURN_KEY ;
WAIT FOR PROMPT "login:" , 20 ;
```

Figure 15: A SmartCom III Learned Script

Hayes modem.

OPEN ACCESS II

Open Access II, from Software Products International (San Diego, CA), is one of those all-in-one packages for the IBM PC family -- spreadsheet, database, word processor, clock-calendar, and communications. The packages we've looked at so far were intended only for communications, so as a last resort their users could always convert to another package if the CBX conversion posed insurmountable problems. But when communications is only an ancillary function of a larger package, switching packages may not be an option.

The communication feature of Open Access II supports direct mode, as well as user-defined modem dialing sequences. But these cannot be used right away. Rather, one must go through an intricate and time-consuming menu-driven configuration session. But after sufficient effort, it can be made to work with the switch, at least in direct mode. Here's how:

Select Communications from the main menu, then select Modem from the communications menu. When it prompts you for a modem name, say NOMODEM. Then select Format and set the desired speed, parity, data, and stop bits. Then go back to the Modem menu and select Miscellaneous Parameters. Be sure to set "Translate modem character" to No. Now go back to the Communications menu and select the Service Menu. Make a new service called, say, CBX. Select the Format menu, and make all the format items agree with the modem formats (if they do not agree, the connection will not work). Now go back to the Service menu and select the Miscellaneous Parameters menu. Several important settings here are: Autodial No, Hangup on Exit No, Ignore CTS Yes, Ignore Linefeed Yes (if the latter is set to No, then the program will send CR LF whenever you type carriage return, and you will not be able to have a successful dialog with ICS). Pop back up two levels to the Communications menu, select Terminal, and now you can interact with ICS. Wasn't that easy?

Of course it wasn't. But preserving the user's investment in an expensive multifunction package can be worth the extra effort. In any case, if adaptation to the CBX's Interactive Call Setup dialog had not proved possible for Open Access II, then it would still have been possible to keep the package, but to use a separate package for communications, or to offer the user the more expensive 244PC ROLMphone alternative.

Another Package, Another Trick

One package we looked at, VsCom, for use between IBM PCs and Wang VS systems, allows direct connections, user-defined dialing sequences, and scripts. Direct connection worked fine on the first try. The evaluation copy of this program came without a complete manual; script file information was not available, nor was a listing of what special characters were allowed in the dialing sequence, for instance to signify pauses. Experimentation did not uncover a pause character. The sample screen showed that "<CR>" stood for carriage return, so the following dialing string was used:

```
<CR>                                                        call myhost<CR>
```

The dialing string has to fit on one line, so the maximum number of spaces were inserted between the initial carriage return and the "call" command. These serve as "padding," or time wasters, to give the switch time to issue its "CALL, DISPLAY OR MODIFY?" prompt before sending the "call" command. Luckily, the switch ignores leading spaces, otherwise this trick would not have worked.

Macintoshes, T-Switches, and DTR

The RS-232-C Data Terminal Ready (DTR) signal is sent from the PC to the ROLMphone to let the CBX know that the PC ready to communicate. This allows software packages to request service by turning on the DTR signal, and to close a connection by turning DTR off. However, certain packages or PCs cannot control their DTR signals. For this reason, the switch's data lines may be configured with "Forced DTR," which means that the switch will pretend the DTR signal is always on, at the expense of software-controlled call establishment and release. The problem is that Forced DTR is not a user-settable parameter. Rather, it must be configured by the CBX management, an administrative procedure that entails a fee, the filling out of forms, and some days of waiting for the request to be processed. Furthermore, the whole concept is intrinsically baffling to users. Therefore, it was decided that all lines should be configured the same, without Forced DTR.

But then what is to be done about devices that can't turn DTR on and off at will? There are two striking examples. First is the Apple Macintosh, which is quite popular at Columbia. But the Mac's serial communications interface is not RS-232-C, but rather an RS-422 device that is *simulating* RS-232-C. This interface does not provide any modem signals, in particular DTR. Second is the ever-popular T-switch (sometimes called an A-B switch): users who have a PC connected to both a ROLMphone and some other device through a T-switch lose their data connection as soon as they switch their device away from the phone.

Clearly, one does not want to submit a "move-and-change" order every time one turns one's T-switch from A to B, or exchanges a PC for a Macintosh. Our solution to this problem was to build special RS-232-C adapters that go between the phone and the user's cable. These loop one of the phone's constant outputs -- either Pin 9 (Positive DC Test Voltage) or Pin 5 (Clear to Send), depending on the phone model -- back into the phone's DTR input. These adapters are called FOTs ("Fake Out Things"), Models A and B, shown in Figure 16. The male end connects to the phone, the female end to the user's cable. The FOT is analogous to a null modem, in that it tricks the phone into thinking a computer is connected. One

```
   Male       Female                                Male       Female

  FG  1----------1  (Frame Ground)
  TD  2----------2  (Transmitted Data)         FG  1----------1
  RD  3----------2  (Received Data)            RD  2----------2
 RTS  4----------4  (Request to Send)          TD  3----------3
 CTS  5----------5  (Clear to Send)           RTS  4----------4
 DSR  6----------6  (Dataset Ready)           CTS  5---+      5
  SG  7----------7  (Signal Ground)                    |
  CD  8----------8  (Carrier Detect)          DSR  6----------6
 DC+  9---+      9  (Test Voltage)                     |
          |                                    SG  7----------7
 DTR 20---+      20 (Data Terminal Ready)              |
  RI 22----------22 (Ring Indicate)            CD  8----------8
                                                       |
                                              DTR 20---+      20
                                               RI 22----------22

      Model A                                           Model B
```

Figure 16: The FOT

might call it a "null computer."

Once we discovered how to connect Macs to the phone, we found that all popular Macintosh packages were able to use the CBX in direct mode, and some also in dial or script mode, depending on the capabilities of the package (see Table 1).

Hard Cases

WilsonLine, a product of the H.W. Wilson Company (Bronx, NY), is a CD-ROM database package used primarily in libraries, which also dials out to a central database service for updated information. It is a turnkey system that performs a unique function in a proprietary manner, and includes a non-customizable communication feature. Only a Hayes 300 or 1200 bps modem can be used. Wilsonline cannot be used for direct computer connections, nor with non-Hayes modems or with data switches. Our libraries depend on this package. For them, the only solution is the costly 244PC, a ROLMphone which emulates a Hayes 1200 Smartmodem.

We also learned that certain "bank at home" products could not be made to work with the CBX, except with a 244PC. Packages like Chemical Bank Pronto, and Citibank Direct Access only work with modems certified and approved by the bank, and the IBM 9751 CBX is not on their list (nor, in all likelihood, are any other data PBX's). But other banking packages like Chase Manhattan Spectrum can be used with "dumb" (nondialing) modems, and therefore also with the CBX.

Summary

Your organization is installing a new digital voice/data switch. If you have any choice in the selection, you should look for features that will allow the popular RS-232-C communications packages to continue working. A Hayes-compatible dialing language would be a good start, especially if it is not an extra-cost option. Also, as we've seen, some kind of call completion indication is essential to the reliable operation of scripts. This can be in the form of a message, or the transition of the Carrier Detect modem signal from low to high.

To prepare users of communication packages for conversion to the switch, first find some way to explain to them why they won't be needing their modems any more. Be positive -- tell them they can take their modems home and dial in to work from there. If the switch claims to provide the Hayes dialing language, test it with a few packages, preferably Smartcom II or III (if any package knows how to exercise a Hayes modem, Smartcom should!). If it works, you're done.

If your switch has its own special dialing language, you'll have to go through the same procedure we did. Conduct a survey of your users to find out what packages they are actually using. Get copies of each package, along with the manuals.

First find out if the package can be used in direct interactive mode. This can be tricky, because most packages are designed to be used with modems. A PC connected directly to a timesharing computer must be a relatively rare phenomenon judging from the scant attention given to this setup by most package manuals. But this is the back door to the data switch, so you must persist till you find it. In packages like Smartcom and Open Access, you may find yourself knee deep in menus before the magic door appears, but in almost every case it's there. Sometimes near the surface, sometimes deeply buried or cleverly disguised.

All those prompts about modem initialization strings, dialing strings, dialing prefixes and suffixes are ruses and red herrings, irrelevant to direct connections. Eradicate them if you can. If you find a configuration item that says "modem type," and one of the choices is "direct" or "no modem," then you're home free. Try the package interactively in this mode, and it should work.

If the package appears not to support direct connections, you might still get it to work with your switch provided you can supply the required RS-232-C modem signals. For example, if the software refuses to communicate unless it receives the Carrier Detect (CD) signal from the modem, you can trick it by feeding its own Data Terminal Ready (DTR) signal back into its CD signal (by connecting pins 8 and 20 together in the PC end of the RS-232-C cable).

Then try the package's dial feature, if it has one. It's nice to get this working, because then the software package's dialing directory can be used to make outgoing data calls through the switch, in the same way it can be used with a modem. This time, try replacing the dial initialization string or prefix with the one used by your switch. In the IBM 9751 CBX case, the initialization string was a carriage return, and the dial prefix was the word "call" followed by a space. The dial suffix, if required, is normally a carriage return. In our case, we needed a pause after the initial carriage return to give the switch time to issue its prompt.

Finally, try setting up a script, if the package has a script language. This will get your turnkey and pushbutton users off the ground, and it will allow any unattended operations to continue unabated. Some script languages are straightforward, others are best understood by a devious mind. In the latter category come scripts of the "expect-send" variety (like Smartcom II), which assume that something must be received before anything can be sent, and scripts that only match single characters, rather than whole strings, from the input stream (like Smartcom II or VTERM II).

Table 1 summarizes the results for the packages we looked at. We found that nearly all of them allow direct connections, and so can be made to work with with IBM's CBX. Getting the CBX's dialing to mesh with the package is somewhat more problematic. But in all cases, it seems that if the package has a script language, *and* it supports direct-mode connections, then scripts can be written to dial out from the CBX. So the news is almost completely good.

Products Mentioned

The IBM 9750 Business Communication System is a product of IBM, 4900 Old Ironsides Drive, Santa Clara, CA 95054.

CrossTalk XVI is a product of Microstuf Inc, 1000 Holcomb Woods Parkway, Roswell, GA 30076. There is now a newer product, Crosstalk Mk.4, from DCA/Crosstalk Communications, same address.

Kermit is available for more than 300 different machines and operating systems from Kermit Distribution, Columbia University Center for Computing Activities, 612 West 115th Street, New York, NY 10025.

Open Access II is a product of Software Products International, 10240 Sorrento Valley Road, San Diego, CA 92121.

PACX IV is a product of Gandalf Data, Inc., US office: 1020 South Noel Avenue, Wheeling, IL 60090.

ProComm 2.2 is a shareware product from DataStorm Technologies, PO Box 1471, Columbia, MD 65205. There is also a newer product, ProComm Plus.

Smartcom II and III are products of Hayes Microcomputer Products Inc, 705 Westech Drive, Norcross, GA 30092.

VsCom is a product of M/H Group, 222 West Adams, Chicago, IL 60606.

VTERM III is a product of Coefficient Systems, 611 Broadway, New York, NY 10012.

WilsonLine is a product of the H.W. Wilson Company, 950 University Avenue, Bronx, NY 10452.

Package	Version	System	Operates with CBX using.... Direct Connect	DIAL Command	Script Language
Crosstalk XVI	3.61	IBM PC	Y	Y	Y
Crosstalk Mk.4	1.01	IBM PC	Y	Y	Y
Dialoglink	1.20	IBM PC	Y	Y	N
Direct Access	---	IBM PC	N	N	N
Hyperaccess	3.11	IBM PC	Y	N	Y
MS-Kermit	2.31	IBM PC	Y	N	Y
Open Access	II	IBM PC	Y	N	N
PC-Plot	3.601	IBM PC	Y	N	N
PC-TalkIII	---	IBM PC	Y	N	N
ProComm	2.2	IBM PC	Y	Y	Y
Pronto	---	IBM PC	N	N	N
Smartcom II	2.2	IBM PC	Y	N	Y
Smartcom III	1.0A	IBM PC	Y	N	Y
Spectrum	2.1	IBM PC	Y	N	N
VsCom	4.5	IBM PC	Y	Y	N
VTERM III	1.2	IBM PC	Y	Y	Y
Wilsonline	1.19	IBM PC	N	N	N
ZCOMM	3/88	IBM PC	Y	Y	Y
MacKermit	0.9	Macintosh	Y	N	N
MacTerminal	2.2	Macintosh	Y	N	N
Microphone II	2.0	Macintosh	Y	Y	Y
Red Ryder	10.0	Macintosh	Y	N	Y
VersaTerm-PRO	2.00	Macintosh	Y	Y	Y
Kermit-65	3.81	Apple II	Y	N	N
Kermit-80	4.09	CP/M systems	Y	N	N
Xmodem et al	---	CP/M, MS-DOS	Y	N	N
CX	---	DECmate	Y	N	N
Easycom	---	DECmate	Y	N	Y
C-Kermit	4E	UNIX, VMS	Y	N	Y
tip	---	UNIX	Y	Y	N
cu	---	UNIX	Y	Y	N
UUCP	---	UNIX	Y	N	Y
Kermit-11	3.58	DEC PDP-11	Y	Y	Y
Kermit-20	4.2	DECSYSTEM20	Y	N	Y
Kermit-32	3.3	DEC VAX/VMS	Y	N	N

("---" means version number not announced or relevant)

Table 1: Package Summary

Glossary

244PC - A four-line data-equipped ROLMphone that includes a Hayes AT command language interpreter.

ASCII - American Standard Code for Information Interchange, ANSI X3.4-1977. The 7-bit code used by most asynchronous terminals and personal computers for character storage and transmission.

CBX - Computerized Branch Exchange, a family of digital voice/data private branch exchanges manufactured by Rolm and IBM.

CR - Carriage Return, ASCII character 13.

CTS - Clear To Send, an RS-232-C signal sent on pin 5 of the standard 25-pin connector by a modem to a computer or terminal, indicating that the computer or terminal has permission to send data to the modem.

DTMF - Dual Tone Multi-Frequency (Touch Tone).

DTR - Data Terminal Ready, an RS-232-C signal sent on pin 20 of the standard 25-pin connector by a computer or terminal to a modem, indicating that the computer or terminal is online.

Host - A computer capable of supporting multiple simultaneous interactive users.

ICS - Interactive Call Setup, the set of prompts and commands provided to the data user by the IBM 9751 CBX.

LDS-125 - A limited distance modem used with the Gandalf PACX IV data PBX.

LF - Linefeed, ASCII character 10.

PACX - The Gandalf Private Automated Computer Exchange, a data switch.

PBX - Private Branch Exchange, a generic term for any device that switches voice or data calls.

PC - Personal computer.

ROLMlink - The proprietary protocol used between the ROLMphone and the IBM 9751 CBX, allowing simultaneous transmission of voice and RS-232-C data over a single pair of telephone wires.

ROLMphone - A digital telephone for use with the IBM 9751 CBX. When equipped with the data option, capable of simultaneous voice and data connection.

Biographies

Christine Gianone is Kermit Administrator of Columbia University's Center for Computing Activities (CUCCA); Frank da Cruz is CUCCA's Senior Planning Officer for Networks, and author of *Kermit, A File Transfer Protocol*. Chris and Frank are also data communications instructors in Columbia's Division of Special Programs, and co-authors of various articles and a forthcoming book on data communications software and protocols.

List of Figures

List of Tables