

PLEASE NOTE:

You must REGISTER this toolkit to receive technical support and update information!

Once you Register:

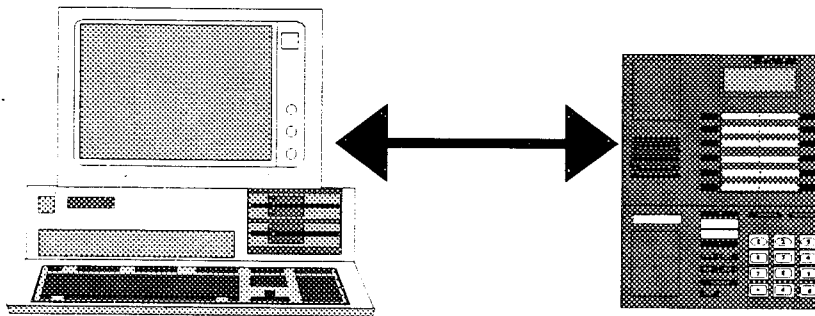
- 1. You will receive our periodic "CTI Newsletter"**
- 2. You can download FREE software OAI 'widgets' from the "OAI BBS".**
- 3. You can contact our "OAI Technical Support" group.**
- 4. You will receive OAI Toolkit and Application note update information.**

**So please take the time to REGISTER NOW.
Please see the end of Section 1 for the Registration Form
and simply FAX or send it to us. It's that EASY.**

Thank You.

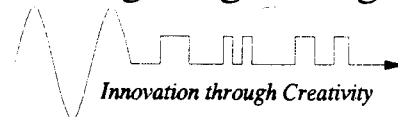
DEVELOPER'S TOOLKIT

ACCESS: Desktop OAI



VERSION: 2.0

Design Engineering



I/T Part No: 550.8201

Access Desktop OAI Toolkit (V2.0)

Desktop OAI Toolkit
SN: 29642042D

What's New in Version 2.0

For those toolkit owners who own a previous version of this Desktop OAI toolkit, version 2.0 has a number of additions or improvements. Even if you have a previous version of the toolkit, you should also take the time register this toolkit as you will then be up-to-date in the OAI Support database and will be allotted more "Level 1" marketing product announcements.

Following is a quick summary of what is new in Version 2.0:

Section 2:

Marketing Program - Is now expanded and more fully explained.

Section 3:

Protocol Document - This new version includes information on Desktop OAI enhancements made in Axxess V3.0

Section 4:

Application Notes - More application notes have been added as well as some have been updated.

Section 5: (and Tools floppy)

DDE Simulator - This tool has been enhanced.

OnLine Help file - An on-line help file containing the entire protocol document is provided.

Section 6: (and Axxessory Connect package)

Axxessory Connect DDE -- New features have been added.

Axxessory Connect TAPI -- The new TAPI service provider API is documented.

Axxessory Connect package - Is the latest version, which includes the TAPI service provider.

Section 7: (and Sample floppy)

Sample Programs - An additional sample program was added.

SECTION 1: Overview

Welcome to the **AXXESS Desktop OAI Developer's Toolkit**. As you are probably aware, the **AXXESS** telephone system is not only a powerful stand-alone telephone system, but also has several "Open-Architecture Interfaces" (OAI) that allow computers to be attached to the phone system to provide even more powerful customer solutions. This toolkit is intended to provide the necessary information and tools that you need to interface with and develop solutions using the **AXXESS Desktop OAI link**.

The **AXXESS Desktop OAI link** is presently available in two versions, both of which are covered by this toolkit. The first version, protocol version V01.02, is now available on **AXXESS V2.x** systems and **AXXENT V1.x**. The second version, protocol version V01.03, is presently available on **AXXESS V3.x** systems. Protocol version V01.03 is a superset of V01.02, so applications developed to work on V01.02 will also work on V01.03. Applications can 'read' the protocol version number from the system to which they are connected and can adapt their feature set accordingly.

Desktop solutions can include use of **Axxessory Connect**, a Microsoft Windows based "PC-Phone" that in addition to providing a graphical user interface for the telephone also provides both a DDE interface and a TAPI service provider interface so that other Windows applications can be seamlessly integrated with the telephone system. Thus, a copy of the **Axxessory Connect** product as well as detailed documentation for the DDE and TAPI interfaces are also provided in this toolkit.

This toolkit will continue to evolve and improve over time and we welcome your input to help us make it better. So please write us or FAX us at the address shown on your 'Registration Form' any time with your suggestions.

Toolkit Contents

This document is organized into sections to make it easier to use as a reference document. The following table shows what is contained in each section:

Section	Description
1	Overview/Registration (this Section)
2	Technical & Marketing Support
3	Protocol Specification
4	Application Notes
5	Development Tools
6	<i>Axxessory Connect</i> (DDE & TAPI)
7	Sample Code

SECTION 1: License

**Inter-Tel Integrated Systems, Inc.
License Agreement**

THE SOFTWARE AND MATERIALS CONTAINED HEREIN IS LICENSED IN ACCORDANCE WITH THE FOLLOWING TERMS AND CONDITIONS. OPENING THE PACKAGE INDICATES YOUR AGREEMENT TO THESE TERMS AND CONDITIONS.

LICENSE:

- (a) Inter-Tel Integrated Systems, Inc. ("Inter-Tel") grants You a non-exclusive, non-transferable license to use the enclosed Inter-Tel software program(s) and accompanying documentation on any one computer machine or Inter-Tel hardware system (which ever applies). You assume the entire responsibility for the selection and installation of the enclosed software program(s) in order to achieve desired results.
- (b) You may copy the Inter-Tel software program(s) contained on diskette(s) for back-up purposes only, provided that You reproduce and place the Inter-Tel copyright notice on each backup copy. You may not copy the Inter-Tel software program(s) contained on any media other than diskette; i.e., hard disk drive, ROMs, PALs, Software Protection Key, etc.

NON-PERMITTED USES:

- (a) You may not use the enclosed program(s) on more than one computer machine or Inter-Tel hardware system.
- (b) You may not sub license, assign or transfer Your rights under the agreement without the prior written permission of Inter-Tel.
- (c) You may not use, copy, alter or transfer, electronically or otherwise, the Inter-Tel software program(s) or documentation, except as expressly allowed in this agreement.
- (d) You may not translate, reverse program, disassemble or decompile the Inter-Tel software program(s).

TERM:

- (a) This license agreement is effective from Your date of purchase and shall remain in force until terminated. You may terminate the license agreement by returning to Inter-Tel the original diskette(s), ROMs, PALs, or other applicable software media and all copies of the Inter-Tel software program(s). The agreement is also terminated if You fail to comply with any term or condition of this agreement. You agree to return to Inter-Tel the original diskette(s) and other applicable software media and all copies of the Inter-Tel software program(s) upon such termination.

WARRANTY:

- (a) Inter-Tel warrants to You that the diskette(s), and/or other applicable software media on which the Inter-Tel software program(s) are furnished are not defective under normal use for a period of ninety (90) days from the date of purchase, as evidenced by a copy of Your sales receipt.
- (b) Inter-Tel and its suppliers' liability and Your exclusive remedy shall be the replacement of any diskette(s) and/or other applicable software media that do not meet the warranty and which are returned to Inter-Tel or an authorized dealer together with a copy of Your paid receipt. THE ABOVE IS THE ONLY WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR USE THAT IS MADE BY INTER-TEL ON THIS LICENSED SOFTWARE. In no event shall Inter-Tel or its suppliers be liable to You or to any third party for consequential, special, indirect or incidental damages which You may incur as a result of using the licensed software, including, but not limited to, loss of data or information of any kind which You may experience. This warranty gives You specific legal rights and You may also have other rights which may vary from state to state. This license agreement shall be governed by the laws of the State of Arizona.

LIMITS OF LIABILITY:

- (a) Inter-Tel and its suppliers' liability (whether in tort, contract, or otherwise; and notwithstanding any fault, negligence, strict liability or product liability) with regard to the Inter-Tel Software Programs or this Agreement will not in any event exceed the compensation paid by You for the Inter-Tel Software Programs.

ENTIRE AGREEMENT:

- (a) This Agreement constitutes the entire agreement and supersedes any and all prior agreements between Inter-Tel and You with regard to the Inter-Tel Software Programs. No amendment, modification or waiver of this Agreement will be valid unless set forth in a written instrument signed by the party to be bound thereby.

DEFINITIONS:

- (a) "You" means, and "Yours" refers to, any person or entity that acquires or uses the Inter-Tel Software Programs. "Computer" means a computer consisting of a single central processing unit, one keyboard and one video display terminal. "Inter-Tel Hardware System" means any proprietary system distributed by Inter-Tel that operates by means of this program(s). "Software" means: the computer programs obtained in this Package (including, but not limited to, codes, techniques, software tools, formats, designs, methods, processes, know-how and ideas); any and all copies, modifications, upgrades, enhancements and new releases thereof made or acquired by You; and any and all manuals and other printed materials contained in this Package.

GOVERNMENT RESTRICTED RIGHTS:

- (a) The Software is provided with restricted rights. Use, duplication, or disclosure by the government is subjected to restrictions set forth in subparagraph c (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 (Oct. 1988) and FAR 52.227-19 (June 1987). Contractor is Inter-Tel Integrated Systems, Inc., Chandler, Arizona 85226.

Inter-Tel Integrated Systems, Inc., Chandler, AZ 85226

SECTION 1: Registration

Please find the registration form on the following page, and fill it out and either mail it or FAX it to us right away. This will allow us to enter your information and serial number into our database so we can 'enable' our technical support for you and also to be able to keep you informed with updated information and related products.

AXXESS: Desktop OAI Toolkit Registration Form

Serial Number:

Desktop OAI Toolkit
SN: 29642042D

First Name:		
Last Name:		
Company:		
Address:		
City:	State:	Zip:
Telephone:	FAX:	
CompuServ:	InterNet:	

OAI Support Billing Method: (Check One)

Credit Card:

Type: VISA Master Charge

Credit Card Number: _____

Expiration Date: ____ - ____

Bill back (AXXESS/AXXENT Dealers ONLY)

I have read, I understand, and I agree to the terms of the license agreement for the above product.

Signed _____ Date _____

To Register -- Mail a copy of, or FAX this page back to:

(NOTE: Always keep the original copy yourself.)

Inter-Tel Integrated Systems, Inc.
OAI Support Dept.
7300 W. Boston Street
Chandler, AZ 85226
FAX: 602-961-6940

SECTION 2: OAI Technical Support

We are committed to helping make your development efforts reach a successful conclusion as quickly as possible. We therefore have designed a *OAI Technical Support Program* to provide you with additional information and access to our best design engineers to help answer your questions and develop your products. **PLEASE NOTE:** We must receive your completed 'Registration Form' before we can provide technical support (see previous section).

The following document describes the OAI Technical Support Program and how to access the OAI Support Group.

OAI Support Overview

The OAI Support Group is available to *Axxess OAI Developers* to provide technical support and access to OAI Design Engineers. This OAI Support is accessible through several electronic mediums (E-Mail and 'Forums') in addition to standard telephone and FAX calls. The electronic 'forums' can be especially useful because developers can access the latest application notes, sample codes, and general information 24-hours/day without directly contacting the OAI Support Group.

When the OAI Support group is contracted, there are two levels of support: 'Basic' (level 1) and 'Advanced' (level 2). The 'Basic' support level is free (no charge) and is always the initial starting point for each new support 'call'. At this 'Basic' level, OAI developers can request product information, report problems, suggest product enhancements, and request information or application notes on existing OAI integrations.

The 'Advanced' support level is billable on a per 'Incident' basis and provides access to the Software Design Engineers within the OAI Support group to help solve the more difficult integration issues/problems and also to provide some direct development resources. Please refer to the following section on "Support Incidents" for more details on these support levels.

Figure 1 illustrates the "typical" flow of a OAI support call.

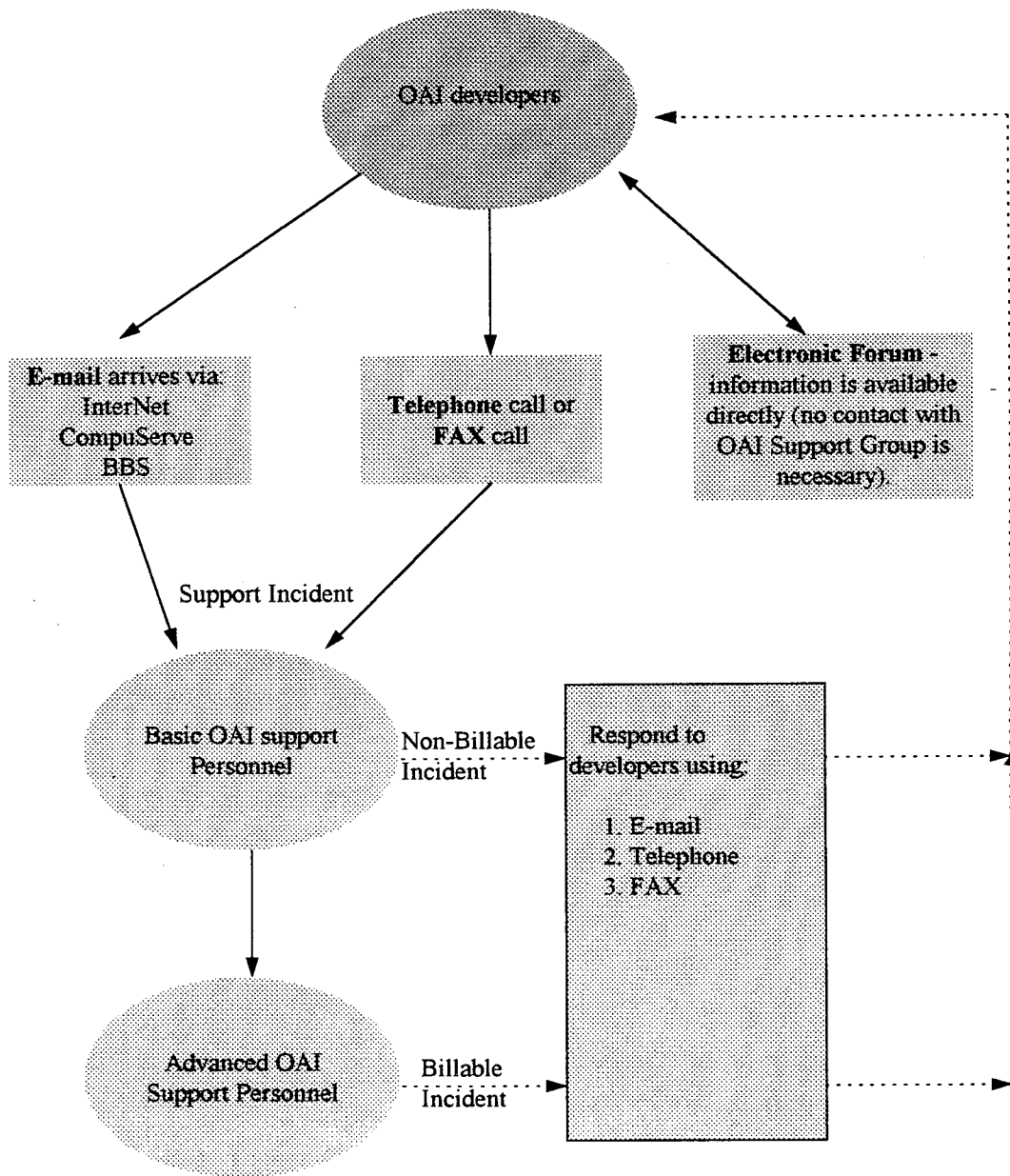


Figure 1

E-Mail / Electronic Mediums

CompuServe

The OAI Support Group maintains a sub-forum for 'OAI developers' on CompuServe. This provides (1) Electronic mail and (2) some file storage areas for on-line perusal and uploads/downloads.

The CompuServe address to send E-mail to the OAI Support and the CompuServe sub-form information will be FAXed to you when your Registration form has been received and processed.

InterNet E-mail

The OAI Support Group will response to E-mail messages on the *InterNet*. These E-mail messages can also have files attached so it is fairly simple to send binary files, word processing documents, or plain ASCII files. Since the *InterNet* is accessible from many other 'On-Line' electronic mail services it provides almost universal access.

The InterNet address for the OAI support group information will be FAXed to you when your Registration form has been received and processed

Inter-Tel BBS

The OAI Support Group will also provide an electronic bulletin board service (BBS) for those not wishing to subscribe to CompuServe or InterNet. This BBS will contain OAI information, including Application notes, sample code, and other relevant OAI information, that can be accessed 24-hours/day seven days/week using a dialup modem. Once your registration form has been received and processed, you will be given an account and access information to reach this BBS.

Telephone & FAX Calls

The OAI Support group can be reached by telephone by calling the current Inter-Tel telephone support phone number 800-669-5858 (or 602-961-9000) and requesting the "OAI Support Group". The OAI Support Group is staffed between 8AM and 5PM Mountain Standard time on normal business days, but when all group members are busy or after hours you will reach a voice mail box where you can leave a detailed message.

The OAI Support group can also be contacted by FAX which can be quite useful for describing problems or attaching printouts. This FAX number is 602-961-6940 and is available 24 hours/day.

Support Incidents

A support incident is a question or a request for assistance from an OAI developer to the OAI Support Group. In some cases these incidents fall under “normal” technical support issues, such as reporting a bug, requesting an enhancement to a product, or requesting additional information about the OAI toolkits. In other cases, these incidents will fall outside the realm of ‘normal technical support’ such as, requesting help on developing a specific OAI application, requesting help on using the various tools in the OAI toolkit, etc. These type of incidences escalate to a higher level, are handled by a “OAI Design Engineer”, and are considered billable.

Following is a description of the two different OAI support levels:

1. **Basic (Level #1)** -- This support level is **free**. OAI developers would use this level to report bugs, submit PERs (Product Enhancement Requests), and request work-arounds for “known” problems.
2. **Advanced (Level #2)** -- This support for this level is **billable** as there will be a charge of \$20 per incidence and is provided by Inter-Tel “OAI Engineers”. At this level Inter-Tel is guaranteed to respond to the customer in 1 working day.

Generally, an “Billable Incident” is considered to be up to 20 minutes of the “OAI Design Engineer’s” effort. Each customer ‘call’ will be limited to one incident unless specifically discussed with and agreed to by the customer in advance (or during the call if it is a ‘live call’). Thus, typically the “OAI Design Engineer” will make a best effort to answer the question or solve the problem in about 20 minutes. If the problem is expected to take considerably more time than 20 minutes, the “OAI Engineer” will give a ‘best efforts’ high-level answer in that 20 minute timeframe including a outline of the estimated steps to achieve a ‘complete solution’ and the estimated efforts.

Problems that turn out to be Inter-Tel product will NOT be considered a “Billable Incident” even if the call has escalated to the “Advanced” level.

Payment Methods for Billable Incidents

The following payment methods are available for “Billable” incidents:

1. **CREDIT CARD** - Your **VISA** or **MasterCard** credit card can be used to pay for “Billable” incidents. To expedite service, it is helpful if this credit card information is provided when the initial “Registration Form” is sent or FAXed into the OAI Support group. If this information is not provided on the “Registration Form”, it will need to be provided on each ‘call’ (E-mail, telephone, or FAX) if ‘Advanced’ service level is to be provided.

2. **Bill-Back Method** - Some **Axxess** dealers will be allowed to be billed back (using the normal Inter-Tel billing cycle/procedures) for their “Billable” incidences.

SECTION 2: Marketing Support

When you have developed a product or integrated an existing product to work with the **AXXESS** system, you may wish to market your product to other existing or potential **AXXESS** customers. We have "**CTI Products Marketing Support program**" to assist all registered **AXXESS** toolkit owners in marketing their products. This marketing program has three different 'levels' of support to give you choices in marketing your product. The highlights of these levels are shown below with more details on the following pages:

Level 1 -- "3rd-Party" Products Category:

- Monthly *3rd-Party Product Announcements* are FAXed to all appropriate Inter-Tel dealers
- *3rd-Party Product Announcements* can be included in this *CTI Newsletter*
- Interested dealers will then contact you, the CTI developer, directly to purchase the product and receive all product support.

Level 2 -- "Partnership" Products Category:

- Proven "Level 1" products can move to this category
- Inter-Tel performs some product testing to insure **AXXESS** compatibility
- Inter-Tel distributes product through "**Inter-Tel Factored Products**" division and product catalog
- Inter-Tel has "accounts receivable" responsibility
- You, the CTI developer, have all product-support responsibilities
- These products can be included in 'Press Announcements' by Inter-Tel
- Opportunities to participate in Inter-Tel product presentations (e.g., Trade Shows, Dealer meetings, etc.)

Level 3 -- "OEM" Products Category:

- Inter-Tel performs some product testing to insure **AXXESS** compatibility
- An Inter-Tel exclusive or 'exclusive version' of the product is created
- Inter-Tel distributes product as an "Inter-Tel product"
- Inter-Tel has "accounts receivable" responsibility
- Inter-Tel has "1st-level" product support responsibility
- Products are included in 'Press Announcements' by Inter-Tel
- Products are demonstrated in Inter-Tel product presentations (e.g., Trade Shows, Dealer meetings, etc.)

Marketing Support -- Level #1 -- "3rd-Party" Products

At marketing level #1 ("3rd-Party Products"), Inter-Tel will help introduce your product to our network of dealers. The Inter-Tel dealers can then contact you directly for detailed product information, to purchase the product, and for all sales and technical support. Inter-Tel does not test or validate your product in any way, and thus does not directly endorse your product. Plus Inter-Tel is not involved in any way with the sales or support of your product.

To request that your product be announced at level #1, you simply fill out a copy of the "CTI Product Announcement Request Form" (see following page) and FAX or mail it to our OAI Support group. At the beginning of every month, Inter-Tel will include the information in the "CTI Product Announcements" that are FAXed to all of our dealers. Interested dealers will typically contact you soon thereafter, and you are on your way.

NOTE: Please keep the original of this form unmarked so you can make a copy whenever you are ready to FAX in another product announcement. You will be limited to 10 of these product announcements for each registered toolkit.

CTI Product Announcement Request Form

I'd like to have my product included in the next Inter-Tel "CTI Product Announcement" to be sent out to appropriate Inter-Tel dealers. I understand that any interested dealers will then contact me directly and I will be totally responsible for selling and supporting the product directly with no involvement from *Inter-Tel Integrated Systems*. Following is information on my product (can be on an attached page).

Signed: _____ Title: _____ Date: _____

Requested Announcement Date:

Contact Information:

Company Name:

Contact Name:

Phone Number:

FAX Number:

Mailing Address:

Product Name:

Product Description: (200 words or less)

Product Pricing:

Product System/Network Requirements:

Inter-Tel Products Used (*i.e. AXXESS V2.0, Desktop/System OAI, etc.*):

Computer Configuration (*CPU type/speed, OS, RAM, Disk, etc.*):

Special Telephone Network Requirements (*i.e. T1, CallerID, etc.*):

Product Status: Beta Released Release date: Restrictions?:

Number of Installed Sites:

Referrals Available?: (Y/N)

Future Planned Enhancements: (*50 words or less*)

Marketing Support -- Level #2 -- “Partnership” Products

When your product has been installed and is running well on a number of installations, you can request that your product be moved to a “Partnership” product and marketed at level #2. Once at this level, your product will be listed in the Inter-Tel Factored Products catalog, and sold by Inter-Tel to the Inter-Tel dealers instead of directly by you. Inter-Tel is thus more involved in sales and marketing of your product, including being responsible for “accounts receivable” from all of the dealers.

To get more detailed information on becoming a “Partnership” product, contact the OAI support group. Typically you should have at least ten verifiably good references of installations of your product, and be prepared to send one of your products to Inter-Tel for validation testing with our telephone systems.

Please note that there are some restrictions on the number and type of products that are allowed to be “Partnership” products.

Marketing Support -- Level #3 -- "OEM" Products

Inter-Tel selects certain products to become "OEM" products when a strong market need has been identified in an area. Inter-Tel becomes even more involved in the sales and support of these products as well as getting involved in the technical support and product development issues.

If you think you may want to have Inter-Tel consider selling your product as an OEM product, please contact our OAI support group for more details. Please note that there are restrictions on the number and type of products will be distributed as "OEM" products.

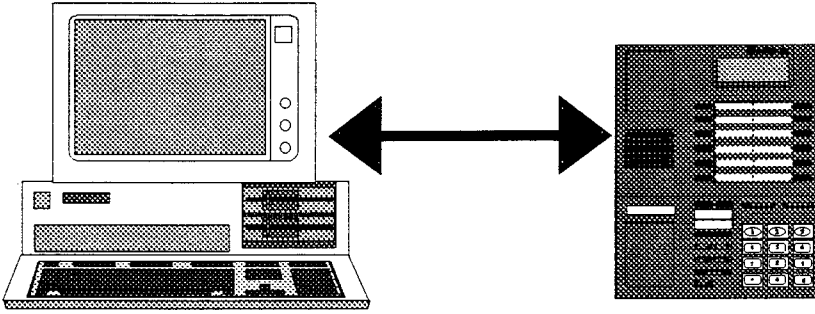
SECTION 3: Protocol Specification

This section contains the detailed protocol specification for the **Axxess Desktop OAI Link**. Because this protocol is a 'superset' of the "Hayes Modem Command Set", many existing software packages that are already capable of dialing calls using a 'Hayes-compatible modem' will automatically work with the **Axxess Desktop OAI link**.

In addition, as you will see when you read the specification, the protocol allows you to develop or enhance application programs to do much more than simply dialing. This would include recognizing whenever a call rings on the attached phone along with the calling party's telephone number (with ANI or CallerID), and looking the phone number up in a 'database' and displaying the appropriate database record on the screen ("Screen Pop" is the buzzword for this feature).

Please note, the protocol information provided in this section is also included as an "On-Line Help" file on the "Tools" floppy (see section 5) so that you can easily search and refer to this information from a standard MS Windows-based PC.

ACCESS: Desktop OAI Protocol



Design Engineering
Innovation through Creativity

Table of Contents

PROTOCOL OVERVIEW	3
COMMANDS	4
OVERVIEW:.....	4
AT - ATTENTION CHARACTERS	5
A - ANSWER COMMAND	5
D - DIAL COMMAND.....	5
E - ECHO COMMAND.....	6
H - HOOKSWITCH COMMAND.....	6
I - VERSION COMMAND	7
L - VOLUME COMMAND	7
M - MUTE TOGGLE COMMAND	8
Z - RESET COMMAND	8
_B - MESSAGE CALLBACK COMMAND	8
_C - CONNECT CALL COMMAND.....	9
_E - EXTENSION KEY COMMAND.....	9
_G - GENERIC OPERATION COMMAND.....	9
_I - INTERRUPT EVENT COMMAND.....	10
_K= - SEND KEY COMMAND	11
_KM - MENU KEY COMMAND	12
_L - LOG SERIAL NUMBER COMMAND.....	12
_M - MODIFY CALL INFORMATION COMMAND	12
_N - MOVE AS 'N'EW	13
_O - CONFERENCE CALL COMMAND	13
_T - TRANSFER	14
_V - DIVERT	14
_X - CANCEL MESSAGE COMMAND.....	15
INTERRUPT RESPONSES	16
OVERVIEW.....	16
CALL STATUS RESPONSES	16
<i>Call Types.....</i>	<i>16</i>
<i>Call Flags.....</i>	<i>16</i>
<i>Volume.....</i>	<i>16</i>
'DEVICE IS BUSY' RESPONSE:.....	16
'CHANGE CALL_ID' RESPONSE.....	17
'CALL CLEARED' RESPONSE:.....	17
'CALL CONNECTED' RESPONSE.....	17
'CALL DELIVERED' RESPONSE	18
'DEVICE IS IN DND' RESPONSE	18
DIALTONE RESPONSE.....	19
'ERROR' RESPONSE	19
EXTENSION STATUS RESPONSE.....	19
'HOLD CALL' RESPONSE.....	21
'INFO DISPLAY' RESPONSE.....	21
LOGGED SERIAL NUMBER RESPONSE	23
'MENU DISPLAY RESPONSE	23
'STATION MESSAGE' RESPONSES	24

'AUDIBLE RING' RESPONSE	24
'VERSION INFORMATION' RESPONSE	25
'WAITING FOR DEVICE' RESPONSE	25
APPENDIX A - AXXESS FEATURE CODES.....	27
APPENDIX B - AXXESS STATION KEY CODES	29

AXXESS: Desktop OAI Protocol

Protocol Overview:

This document describes the AXXESS Desktop Open-Architecture-Interface (OAI) protocol that is provided on the RS232 port of an AXXESS keyset (using a PCDPM). This protocol is extended from the standard 'Hayes-Compatible' protocol and is intended to allow a PC to be attached to a keyset and provide the communications to support features like: PC-based phone, Operator's console, ACD supervisor screen, Integrated Contact Management, etc.. And, since the protocol is 'Hayes compatible', most off-the-shelf application programs that support dialing using a 'standard' Hayes-compatible modem will work well without any modification.

This protocol is purposely kept simple to allow rapid implementation and easy integration by existing and future VARs, and MIS or engineering personnel. It supports 'traditional' Hayes-type commands and 'Inter-Tel Extended' commands which are initiated by the attached PC, but it also supports 'Interrupt Events' which originate elsewhere in the system and are sent to the attached PC.

The keyset PCDPM supports standard RS232C with no parity, 8 data bits, 1 stop bit, and at a baud rate of up to 4800 (the baud rate is programmable from the keyset via feature code: 393) The attached PC must be set up to support hardware handshaking on its transmit side to avoid overrunning the phone system. Also, the application must be able to receive characters as fast as the system sends them because hardware handshake is not honored in this direction, so the baud rate should be set accordingly.

Since the PCDPM doesn't support end-to-end data connections, this protocol always stays in the 'command state' (and never switches to the 'data state'). Generally, commands sent will be as if they were keys pressed on the keyset. So, for example, if an "Off-Hook" command is sent, it will be as if the keyset went 'off-hook' on its speakerphone. Dialing a 'Flash' is as if the keyset SPCL key is pressed, so the digits following would be interpreted as a feature codes. (Please see Appendix A for a list of the AXXESS feature codes).

Commands

Overview:

The term 'application' is to refer to the program that issues the commands, while the term 'device' refers to the object that receives the commands and issues the responses. It is recommended that the 'application' send the Reset Command as its first command to reset the port to its default mode. Please refer to the Reset Command ('Z').

All commands must start with the 'AT' character sequence (any characters preceding this 'AT' sequence on a command line will be ignored). After the 'AT' sequence, up to 80 additional characters will be processed. The command ends with either a <CR> or a control-X input. All commands are case insensitive.

When the command ends, an immediate '*command-syntax*' response is issued: which is either an "OK"<CR><LF> if the command is correct and supported or an "ERROR"<CR><LF> response. An "OK"<CR><LF> response is always issued when a control-X input (abort this command) is received.

A deadman timer (30-seconds) is started once the first letter (the 'A') of an 'AT' command is received. If the terminating <CR> is not received before the deadman timer expires, a Control-X input is assumed and the command is terminated.

The protocol supports interrupt events that, when enabled, are sent to the port to notify the application of call activities. These interrupt events are Disabled whenever the port is defaulted and must be specifically enabled when they are desired. Once the first letter ('A') of a command has been entered, all interrupt events are suspended and queued up until the command has been completely entered (or aborted) and the '*command-syntax*' response has been issued. Please see the 'Interrupt Event' command ('I').

Some commands and responses have a '*Call_ID*' parameter. A '*Call_ID*' is an ASCII string (includes the following possible characters: 0123456789@#*PFX) assigned by the phone system to each call, which provides a reference identifier so that multiple simultaneous calls can be managed on a phone. For example, the phone could have one call in the 'talking' state, another 'On-Hold', and two more calls 'Ringing In'.

On some commands and responses, text strings are used as parameters. These text strings need to be enclosed in vertical slash characters to allow use of any other visual ASCII character within the text string.

Commands Supported

AT - Attention Characters

Parameters: None
Response: "OK"<CR><LF>

This command is a "No-Operation" command but is useful to test the connection and as a periodic command to be used to keep the 'application halted timer' from expiring on advanced applications. The AT characters should also precede any other command.

A - Answer Command

Parameters: None
Response: "OK"<CR><LF>

This command causes the device to perform the ANSWER feature. The Answer feature connects the device to a particular call that is present at the device, according to a predetermined algorithm. This call may be ringing or holding at the device.

Example:

Answer the next call: "AT A"

Call Status Event responses will commence if a call is accessed. Be sure that Call Status Event responses are enabled; please refer to the Interrupt Event (I) command.

D - Dial Command

Parameters: A string containing one of the following options followed by dialing digits:

Options:

A - Alphanumeric text	(required)
T - touch tone	(implied)
P - pulse	ignored and mapped to 'T'

Digits:

0-9, *, #	Keypad digits
, (Comma)	Pause digit
! (Exclamation Point)	Hookflash (SPCL key)
- (Hyphen)	Ignored
() Parenthesis	Ignored
; (Semi-Colon)	Ignored
Space	Ignored
[0x20...0x7d]	ASCII Characters

Default: None
Response: "OK"<CR><LF>

This command instructs the device to dial the specified digit string. This string may represent an internal extension number, an outside number, a feature code, or any combination of these elements. The main point is that these digit string inputs are treated as though they were physically generated at the device by pressing the equivalent keys.

In touch-tone mode (option 'T'), the inputs (0-9, *, #) are treated as though they are physical keypad inputs generated by pressing the corresponding digit on the keypad of the device. The hookflash input (!) is treated as though it is a physical SPCL key input; the SPCL key input allows access to the entire feature code set of the system (refer to Appendix A for a list of feature codes). The pause input (,) causes a pause to be inserted into the dial string.

Example:

Here is the command to dial the number '961-9000': **"AT DT 9619000"**

In alphanumeric text mode (option 'A'), all inputs are treated as a literal ASCII text. All printable ASCII characters in the range of 0x20 to 0x7D, inclusive are accepted. When using the alphabetic text mode, it is recommended that the application perform all local string editing.

Example:

Put present call on HOLD: **"AT D!336"**

Put phone in DND "OUT OF TOWN 'TIL" (message #11) "FRIDAY 7/27":

"AT D!370 11"

"AT DA |FRIDAY 7/27|"

Various event messages will commence if the logical key initiates an action that generates Event messages. Be sure that the appropriate Event responses are enabled; please refer to the Interrupt Event ('_I') command.

E - Echo Command

Parameters:	E0 - disables the input echo option. E1 - enables the input echo option. (Default)
Default:	None
Response:	"OK"<CR><LF>

This command controls the duplex of the application inputs. When the input echo option is ENABLED, the device will echo all application inputs at the input display device. When the input echo option is DISABLED, the device does not echo application inputs.

Example:

Turn off Echo: **"AT E0"**

This command does not affect Response Data or Event messages nor does this command trigger any Event messages.

H - Hookswitch Command

Parameters:	H0 - ignored H1 - performs a logical Off hook transition. H2 - performs a logical Off hook transition. H3 - performs a logical ON hook transition.
Default:	None
Response:	"OK"<CR><LF>

This command manipulates the hookswitch of the device. The logical ONHOOK transition terminates the current activity and returns the device to idle; if the device is PHYSICALLY offhook, the device returns to dial tone. The logical OFFHOOK transition issues Dial Tone to the device if no calls are ringing; if a CO call is

ringing AND Auto CO Answer Mode is ENABLED, or if an IC call is ringing AND Auto IC Answer Mode is ENABLED, logical OFFHOOK transition connects the device to the ringing call.

The 'H0' command is specifically ignored because it is considered a "Modem" On-Hook command and the device is not a modem. Many application programs that use "Modem Dialing" will send this command to hang up the Modem after the handset on the attached phone has been picked up.

Example:

Hang-up (go ON hook): "AT H3"

Call Status Event responses will commence if a call is accessed. Be sure that Call Status Event responses are enabled; please refer to the Interrupt Event ('I') command.

I - Version Command

Parameters: I0 - requests Protocol Software version information.
I1 - requests Keypad version information.
I2 - requests KSU Software version information.

Default: None

This command is useful for checking various versions to allow the attached program to check & maintain compatibility across version software versions and keypad types. A 'Version Information' Event will be sent as a response to this command.

Example:

Request Protocol Software version: "AT I0"

L - Volume Command

Parameters: L0 - save volume for current volume mode.
L1 - active volume level 1 for the current volume mode.
L2 - active volume level 2 for the current volume mode.
...
L8 - active volume level 8 for the current volume mode.

Default: None

Response: "OK"<CR><LF>

This command controls the volume level of the current volume mode active at the device. A volume mode is any one of the user programmable volume modes available at the device (e.g., HANDSET IC, CO or TONE; HANDSFREE IC, CO, or TONE; BGM, or ALERT). When the parameter is (0), this command saves the volume level for the current volume mode. This saved volume level is then preserved for all subsequent uses of the current volume mode.

When the parameter is (1-8), this command specifies the volume level to active for the current volume mode. If the volume level is valid for the current volume mode, that specified volume level is established. If the volume level is invalid, the maximum volume level allowed is established and a volume limit tone will be issued on the device.

Example:

Set to Minimum volume: "AT L1"

This command does not trigger any Event messages.

M - Mute Toggle Command

Parameters: M3 - toggle microphone mute status.
Response: "OK"<CR><LF>

This command toggles the Microphone Mute status of the device. When the Mode is ENABLED, the device is in RECEIVE-ONLY audio mode. When the Mode is DISABLED, the device is in TWO-WAY audio mode.

Example:

Toggle the Microphone Mute: "AT M3"

This command does not trigger any Event messages.

Z - Reset Command

Parameters: None
Response: "OK"<CR><LF>

This command resets the command interface of the device and defaults all options. It also alerts the device that an application is present. This is the first command that the application should issue to the device when it establishes a connection to the device. It is also recommended that this is the last command an application sends.

The Reset Command sets the following defaults:

- enable the input echo option.
- disables all interrupt event messages.

Example:

Reset command interface "AT Z"

This command does not terminate ANY calls at the device. This command does not trigger any Event messages.

_B - Message CallBack Command

Parameters: <Extension Number> - (optional) A string of one to five ASCII characters that represent the extension number of the device whose message call is being returned. If this parameter is not provided, the current message is assumed.
Mailbox Number - (optional) If the Extension Number is a voice mail notification extension, this parameter directs the callback to a specific mailbox. If this parameter is not provided, the default is the current voice mail message (only available on protocol V01.03 or later).
Response: "OK"<CR><LF>

This command instructs the device to call back the Extension that left a STN Message. Using this command (instead of a simple "ATDT") lets the system know that the call is a return call from a message that was left which allows the call to be handled appropriately by the system.

Example:

Call back Message from Extension 105: "AT_B 105"

Call back to get Voicemail (Ext 299) from Mailbox 1201: "AT_B 299,1201"

_C - Connect Call Command

Parameters: <Call_ID> of the call to be connected. If no Call_ID is specified, the next call will be accessed.
Response: "OK"<CR><LF>

This command allows the application to access a particular call, regardless of its status, priority, or sequence number. The call with the specified call ID is connected to the device. If no call ID is specified, the next call is connected to the device. If an invalid call ID is specified, no action takes place.

Examples:

Connect to Call ID @004: "AT_C@004"

Connect to Next Call: "AT_C"

Call Status Event responses will commence if a call is accessed. Be sure that Call Status Event responses are enabled; please refer to the Interrupt Event ('I') command.

_E - Extension Key Command

Parameters: <Extension> = Extension number 'under' the key
Response: "OK"<CR><LF>

This command allows the application to simulate a key being pressed. Since, on the AXXESS system an 'Extension' can be a feature code, a trunk access code, a station extension, etc., this command provides a means to simulate DSS keys, trunk keys, feature keys, etc.

Depending on the extension number sent and the state of the keyset, this command could result in appropriate Event messages being sent.

Example:

Simulate pressing DSS key for Extension 100: "AT_E 100"

_G - Generic Operation Command

Parameters: <Operation> = ID for operation to be performed (3-digit code)
..... = Other parameters as needed
Response: "OK"<CR><LF>

This command allows the application to access a particular 'generic' operation at the device. Since, not all operations on the AXXESS system have an associated feature code, this command provides a means to access features not covered by feature codes.

This command may generate event responses, depending on the specified operation and the event responses being enabled.

<u>Operation</u>	<u>Operation ID</u>	<u>Extra Parameters</u>
Divert Ringing Call to DND	000	none

NOTE: Using this operation will send the ringing call to DND and leave the telephone in DND state.

Divert Ringing Call to Voice Mail	001	none
Call Message Center	002	none
Call Voice Mail	003	none
Cancel DND	004	none
Cancel FWD	005	none
Leave Voice Mail	006	none
Divert Ringing Call to Specific DND	007	<DND Msg #>, <DND Custom Text >

where:

DND MSG # - (optional) This parameter specifies the number of the DND Message. If this parameter is not present, or if this parameter is out of range, the system uses DND MSG #01 by default. Valid DND message numbers are 1 to 20, inclusive.

DND Custom Text - (optional) This parameter specifies the custom DND text string for the DND message. This string can be up to 16 characters in length.

NOTE: Using this operation will send the ringing call to the specific DND message but will NOT put the telephone in DND state. This operation code (007) is only available on protocol V01.03 or later.

Examples:

1. Divert call to DND: "AT _G 000"

2. An IC call is ringing in and you want to tell the caller that you will back after 3:30PM (assume DND MSG #15 is programmed to "I'LL CALL YOU"):

_G 007,15,|AFTER 3:30PM|

_I - Interrupt Event Command

Parameters: <Interrupt Mask> - An ASCII string that represents the hexadecimal value of the interrupt mask.

Response: "OK"<CR><LF>

This command controls the interrupt event mask of the device. When a mask bit is SET, the associated interrupts events are enabled. When a mask bit is CLEAR, the associated interrupts events are disabled. A string of "0" (0x00) will disable all interrupt events; a string of "FFFF" (0xffff) will enable all interrupt events. NOTE: If less than 4-digits are sent, leading zeroes are assumed. This command does not trigger any Event messages; it merely enables and disables the generation of different types of Event messages.

Once any interrupt events are enabled, the attached application must submit at least one command every 4.5 minutes ('application halted' timer), otherwise the system will assume that the application has been disconnected and will disable all interrupt events until they are re-enabled with this command again. This is necessary to avoid excessive RS232 traffic to non-existent or disconnected applications.

The interrupt mask of the device is a 16-bit field defined as follows:

Bit:	15- msb	14	13	12	11	10	9	8
------	---------	----	----	----	----	----	---	---

Events:	-	-	-	x	x	x	x	x
Use:	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved

Bit:	7	6	5	4	3	2	1	0 - lsb
Events:	-	-	-	x	x	x	x	x
Use:	Own Keypad Status	Menu Displays	Info Messages	Station Messages	Reserved	Extension Monitoring	DSS Monitoring	Basic Call Events

Example:

Enable 'Own Keypad Status', 'Extension Monitoring' and 'Basic Call Events': "AT_I0085"

NOTES:

1. Basic Call Events (bit 0) - When this is enabled, Basic Call event messages will be sent for all calls made/received by the attached keypad.

2. DSS Monitoring (bit 1) - When this is enabled, Extension status messages will be sent for each station/trunk programmed under a DSS key on a DSS instrument. This will allow monitoring of status of all phones in the system. NOTE: There is a system limit to the number of simultaneous real or 'virtual' DSS instruments that it will support. If that limit has been reached, DSS Monitoring will not be allowed to be enabled and an 'Error' (ER) response will be returned.

3. Extension Monitoring (bit 2) - When this is enabled, an Extension status message will be sent each time the status changes on any monitored extension. Monitored extensions are those extensions under 'DSS' keys resident on the attached keypad.

4. Station Messages (bit 4) - When this is enabled, Station Message responses will be sent when 'station messages' arrive on or are deleted from the attached keypad.

5. Info Messages (bit 5) - When this is enabled, Info response messages will be sent as if a 2x16 Info display is attached.

6. Menu Displays (bit 6) - When this is enabled, MENU response messages will be sent as if a 4x16 Menu display is attached.

7. Own Keypad Status (bit 7) - This mask bit turns on events to be sent whenever the attached keysets status changes.

8. Extension Status - Depending on the system size, configuration, and activity, it is possible that at times the system could become so overloaded with messages that it may not send all transitional messages for Extensions. That is, for example, if during a busy time for the system an extension went off hook (BUSY) and right back on hook, the BUSY Response for that extension may not be transmitted. Therefore, attached application programs should NOT assume that they will receive responses for every extension transition.

_K= - Send Key Command

Parameters: Two ASCII characters that represent the two digits of a logical key number.
Response: "OK"<CR><LF>

Example: "51" => Dec 51 => log key 51

This command allows the application to simulate pressing any physical key on the device. Each physical key on the device has a logical key number. These logical key numbers are uniform across various device keymaps. It is recommended that this command not be used by 'normal' applications and is actually included for use by 'test' and 'debug' applications. For a complete list of the logical key numbers please refer to Appendix B (Axxess Station Key Codes).

Example:

Simulate the user pressing the "Outgoing" key: "_K=31"

Various event messages will commence if the logical key initiates an action that generates Event messages. Be sure that the appropriate Event responses are enabled; please refer to the Interrupt Event ('_I') command.

_KM - Menu Key Command

Parameters: <Key Number> - The number of the menu key pressed (1 through 8).
Response: "OK"<CR><LF>

This command instructs the device that the user has pressed a menu key. This is used in conjunction with the "Menu Display" response.

Key Number: (1 - 8) with odd numbers on left and even numbers on the right as follows:

①	②
③	④
⑤	⑥
⑦	⑧

Example:

Press menu key #2: "_KM 2"

_L - Log Serial Number Command

Parameters: <serial number> - can be an Alpha/Numeric string of up to 8 digits
Response: "OK"<CR><LF>

This command is a request to the system to see if this <serial number> is already in use in the system and if not log it as in use at this extension. It will then remain in use at this extension until the port is reset (Z - command), a port 'deadman timer' timeout occurs, or a different serial number is logged at this port. This command is useful for attached software programs to check to see if illegal duplicate versions of software are in use. This command always generates a "Logged Serial Number" (LS) response.

Example:

Log Serial number 127AX55: "_L |127AX55|"

_M - Modify Call Information Command

Parameters: <Call_ID> - (if not provided, then it affects the present Call)
<Caller Number> - optional
<Caller Name> - optional text string
<Account Number> - optional Account Number (up to 12 digits)
Response: "OK"<CR><LF>

This command instructs the device to modify the caller information associated with the specified call. The caller information consists of the caller's name and the caller's number. The name and number fields are optional. Once a name or number is updated, the system uses the updated values for the duration of the call (in call Transfers, SMDR, etc.). This command is very useful for attaching "CallerID" information to a call that doesn't already have that information or has incorrect information (i.e. the caller is actually calling from a pay phone instead of his business phone). Each field should be separated with a comma.

Examples:

Substitute following info on the present Call : "AT_M , 5551212, |BOB JONES|"

Add/Change account number of call #5: "AT_M 5, , , 123456"

This command does not trigger any Event messages.

_N - Move as 'N'ew

Parameters: <Call_ID> = Call_ID of the call to be Moved as New.
 <To_Extension> = Extension number of device to receive the New call
 <Number> = Outside phone number (if applicable), can include pauses ('P' and
 flashes ('F'))
 <From_Extension> = optional parameter - Extension number of device to move
 the call from. If this parameter is omitted, the 'attached device' Extension is
 assumed.
 <Vmail Parameter> = optional parameter - If the call is moved to V-Mail (or is
 eventually forwarded to V-Mail, then this parameter is passed to the V-Mail unit.
 Typically this would be set to the Voice Mailbox number that should receive the
 call. This parameter is limited to length of 5 digits of type: '0-9', '#', '*' or 'P'.

Response: "OK"<CR><LF>

This command Moves a call that is presently alerting the attached device (or optionally a different device), thus moving the call to another device or outside phone and makes/keeps the call a "N"ew call. This command is useful to send a call to another phone as if it rang directly in on that phone so the LCD will display a "new call" and if that phone is forwarded to V-Mail, it will go to that phone's V-Mail box. This is slightly different than the 'Divert' command, which acts like the call was Forwarded. If the 'Move' command is successful, a 'Call Cleared' response will be sent (if enabled).

NOTES: You can only 'Move' a ringing (alerting) call. However, the system will not allow you to 'Move' a call ringing into a " Hunt Group" (or "ACD group"). This command is only available protocol V01.03 or later.

Examples:

Move call from this phone to Ext. 100: "AT_N @003, 100"
 Move call from this phone to an outside number: "AT_N @005, 9, 9619000"
 Move call ringing on Ext. 103 to Ext 120: "AT_N @007, 120, , 103"
 Move call ringing on Ext. 103 to Vmail (x299) main greeting: "AT_N @007, 299, , 103"
 Move call ringing on Ext. 103 directly to Vmail (x299) mailbox 105: "AT_N @007, 299, , 103,105"

O - Conference Call Command:

Parameters: Number of Other Parties
 <Call_ID #1> - Call ID of 1st call to be connected into CNF

 <Call_ID #n> - Call ID of 'n'th call to be connected into CNF

Response: "OK"<CR><LF>

This command is used for setting up a conference call with a single command which allows the user interface for setting up a conference to be simplified (versus use of the Conference feature code). If this command is

successful (and if 'Basic Call Events' are enabled), the system will send a 'Call Connected' response with a new <Cnf_Call_ID> with a CNF_Call type.

Example:

Set up a conference call with 3 other parties (to form a 4-party conference):

"_O 3, @103, @137, @199"

_T - Transfer

Parameters: <Call_ID> = Call_ID of the call to be Transferred (if omitted then it refers to the present call - required if <From_Extension> is included).
<To_Extension> = Extension number of device to receive the call
<Number> = Outside phone number (if applicable), can include pauses ('P' and flashes ('F'))
<From_Extension> = optional parameter - Extension number of device to Transfer the call from. If this parameter is omitted, the 'attached device' Extension is assumed.
<Vmail Parameter> = optional parameter - If the call is being transferred to V-Mail (or if it is eventually forwarded to V-Mail), then this parameter is passed to the V-Mail unit. Typically this would be set to the Voice Mailbox number that should receive the call. This parameter is limited to length of 5 digits of type: '0-9', '#', '*' or 'P'. This parameter is only available on protocol V01.03 or later.

Response: "OK"<CR><LF>

This command 'Transfers' a call that is presently 'talking' (connected) on the attached device (or optionally a different device), thus moving the call to another device or to an outside phone. If the 'Transfer' command is successful, a 'Call Cleared' response will be sent (if enabled).

NOTE: The 'Transfer' command works only on 'talking' calls, where the 'Divert' and 'Move' commands work on Ringing calls.

Examples:

Transfer this call from this phone to Ext. 100: "AT_T, 100"

Transfer this call to an outside number: "AT_T, 9, 9619000"

Transfer a call on Ext. 103 to Ext 120: "AT_T @007, 120, , 103"

Transfer a call to Voice Mail (Ext. 299) directly to Mailbox 104: "AT_T @007, 299, , ,104"

_V - Divert

Parameters: <Call_ID> = Call_ID of the call to be diverted.
<To_Extension> = Extension number of device to receive the diverted call
<Number> = Outside phone number (if applicable), can include pauses ('P' and flashes ('F'))
<From_Extension> = optional parameter - Extension number of device to divert the call from. If this parameter is omitted, the 'attached device' Extension is assumed.
<Vmail Parameter> = optional parameter - If the call is diverted to V-Mail (or is eventually forwarded to V-Mail), then this parameter is passed to the V-Mail unit. Typically this would be set to the Voice Mailbox number that should receive the call

call. This parameter is limited to length of 5 digits of type: '0-9', '#', '*' or 'P'. This parameter is only available on protocol V01.03 or later.

Response: "OK"<CR><LF>

This command diverts a call that is presently alerting the attached device (or optionally a different device), thus moving the call to another device or outside phone. This is useful to send a call to Voice Mail or another person if the attached device/person is too busy to take the call. The call will appear the same as it would if the phone system had forwarded the call so the LCD will appear like: "Call FWDed from...". If the 'Divert' command is successful, a 'Call Cleared' response will be sent (if enabled).

NOTES: You can only 'Divert' a ringing (alerting) call. And, unlike the 'Move' command, you CAN 'Divert' a call ringing into a "Hunt Group" (or "ACD group").

Examples:

- Divert call from this phone to Ext. 100: "AT_V@003,100"
- Divert call from this phone to Voice Mail (ext 299): "AT_V@004,299"
- Divert call from this phone to an outside number: "AT_V@005,9,9619000"
- Divert call ringing on Ext. 103 to Ext 120: "AT_V@007,120,,103"
- Divert call ringing on Ext. 103 to Voice Mail (x299) mailbox 105: "AT_V@007,299,,103,105"

X - Cancel Message Command

Parameters: <Extension Number> - (optional) A string of one to five ASCII characters that represent the extension number of the device whose message call is being canceled. If this parameter is not provided, the current message is assumed.
<Mailbox Number> - (optional) If the Extension Number is a voice mail notification extension, this parameter directs the message cancel for a specific mailbox. NOTE: This only removes the message display from the phone, it does NOT remove or delete the message from the voice mailbox. If this parameter is not provided, the default is the current voice mail message (only available protocol V01.03 or later).

Response: "OK"<CR><LF>

This command cancels all station messages left by the specified extension on the attached device.

Example:

Cancel Message from Extension 105: "AT_X 105"

Station Message Event responses will commence. Be sure that Station Message Event responses are enabled; please refer to the Interrupt Event ('I') command

Interrupt Responses

Overview:

Interrupt responses are only sent if they are specifically enabled (using the '_I' command). These responses contain the information necessary for the application to maintain track of the calls present at the device and they are sent as they happen. Responses are queued up (not sent) while any command is being sent from the application. After a command is completely entered and the 'command syntax' response is sent, then any pending responses will be sent.

Blank response fields are always terminated by a trailing comma (","), unless the field is the last field of the response, in which case the field is simply omitted. All response values are in the range of printable ASCII characters, and all responses are terminated with a <CR><LF> sequence.

Call Status Responses:

Call status responses are generated for all calls that appear at the device whenever the call makes an important state transition. Any one call may generate numerous Call Status messages.

The following field values are used in many of the call status responses:

Call Types:

The call type represents the type of the call using the following ASCII representations:

"0x" = 2-way audio connection	+	"0" = IC Call
"1x" = Receive-only audio connection		"1" = CO Call
"2x" = Transmit-only audio connection (*future)		"2" = CNF Call

Call Flags:

The call flags specify any important auxiliary information about the call. These call flags are an ASCII string that represents the hexadecimal value of the event flag mask:

Example: "9C" => Binary 1001 1100

Bit:	7	6	5	4	3	2	1	0 - lsb
Flags:	-	-	-	-	x	x	x	x
Use:	Reserved	Reserved	Reserved	Queue Callback	Hold Recall	Transferred Recall	Transfer Call	Forwarded Call

Volume:

The volume level setting and range for the speaker is often sent as a parameter to allow the attached application to display the setting. The format for the 'Volume' parameter is a 3-digit hexadecimal number consisting of the following:

- 1st digit - Minimum volume setting (usually 1)
- 2nd digit - Present volume setting (0 to F)
- 3rd digit - Maximum volume level setting (0 to F)

Example:

A typical value would be "138", where Volume level 1 is the lowest the setting can go, Volume level 3 is what it is currently set to, and Volume level 8 is the highest the setting can go for this connection.

'Device is Busy' Response:

This response is issued when the specified call, which has been initiated by the device, accesses a device that is busy. This response is only issued when 'Basic Call Events' are enabled.

"BS," <Call_ID>,<type>,<name>, <number>,<volume>

The fields of the response are defined as follows:

Call_ID:	the logical ID assigned to the call.
type:	the type of the call.
name:	Name of other party (text string)
number:	Number of other party
volume:	Volume level of Busy Tone

Example:

IC Call @137 is to a device that is 'Busy': "BS, @137,0,|NORMA|,100,138"

'Change Call_ID' Response:

This response is issued when the phone system has changed the Call_ID of a call (which can occur on certain conditions like a call 'Merge'). This response is only issued when 'Basic Call Events' are enabled.

"CH," <Old Call_ID>, <New Call_ID>

Example:

Call @201 has been changed to Call @137: "CH, @201, @137"

'Call Cleared' Response:

This response is issued when the specified call is removed from the device (this device or the other device hung up on the call). This response is only issued when 'Basic Call Events' are enabled.

"CL," <Call_ID>

The fields of the response are defined as follows:

Call_ID:	the logical ID assigned to the call.
----------	--------------------------------------

Example:

The device on call @137 has now hung up: "CL, @137"

'Call Connected' Response:

This response is issued when the specified call is connected (active) at the device. This response is only issued when 'Basic Call Events' are enabled.

"CX," <Call_ID>,<type>,<name>, <number>,<volume>,<time>,<cost>,<rate>

The fields of the response are defined as follows:

Call_ID:	the logical ID assigned to the call.
type:	the type of the call.

name: Name of other party - text string - If CO = Caller_ID name,
 If IC call = username, If unknown it's blank
 number: Number of other party: If Outgoing CO = dialed number, If Incoming CO =
 Caller_ID or ANI number, If IC call = Extension number, Otherwise - blank
 volume: The volume level setting of the call
 time: if the call is a CO call, this field is present and it
 specifies the elapsed call time (in Seconds)
 cost: if the call is a CO call, this field is present and it
 specifies the elapsed call cost (in Cents)
 rate: if the call is a CO call, this field is present and it
 specifies the current cost rate for the call (in Cents per minute)

Example:

Call @201 (a CO call) is now connected (2-way talking) on the phone at a volume setting of 3 at a rate of 20 cents/minute and has already be connected (somewhere in the system) for 5 minutes and 15 seconds and an elapsed cost of \$1.25: "CX,@201,1,|JANE JONES|, 5551234, 148, 315, 125, 20"

'Call Delivered' Response:

This response is issued when the specified call, which has been initiated by the device, is now 'ringing' at the destination device. This response is only issued when 'Basic Call Events' are enabled.

"DE," <Call_ID>,<type>,<name>, <number>,<volume>

The fields of the response are defined as follows:

Call_ID: the logical ID assigned to the call.
 type: the type of the call.
 name: Name of other party (text string)
 number: Number of other party
 volume: The volume level setting of the Ring Back tone

Example:

An IC Call from NORMA at Extension 100 is 'Ringing' at the device: "DE,@137,0,|NORMA|,100,158"

'Device is in DND' Response:

This response is issued when the specified call, which has been initiated by the device, accesses a device that is in Do-Not-Disturb (DND). This response is only issued when 'Basic Call Events' are enabled.

"DN," <Call_ID>,<type>,<name>, <number>,<volume>,<DND MSG>,<DND text>

The fields of the response are defined as follows:

Call_ID: the logical ID assigned to the call.
 type: the type of the call.
 name: Name of other party (text string)
 number: Number of other party
 volume: The volume level setting of the DND tone
 DND MSG: the canned DND text string that pertains to destination of
 the call.

DND text: the custom DND text string that pertains to destination of the call.

Example:

IC Call @137 is to a device that is in DND - "OUT TO LUNCH", "BACK AT 1:15PM":
"DN, @137,0,|NORMA|, 100, 138,|OUT TO LUNCH|, |BACK AT 1:15PM|"

Dialtone Response:

This response is issued whenever the system issues dialtone to the device or removes dialtone from the device AND when 'Basic Call Events' are enabled.

"DT,"<On/Off>

The fields of the response are defined as follows:

<On/Off>: On = 1, Off = 0

Example:

Dial Tone ON: "DT,1"

'Error' Response:

This response is issued when the system has detected an error relative to the OAI link. An <error code> number is sent as a parameter to 'explain' the reason for the error.

"ER, <error code> "

Error Code Values:

000 = Data Lost - This response is issued when the system output queue (minimum size of 512 bytes) has backed up and overflowed because data is being generated faster than it is being received by the attached application, and indicates that some responses have been lost due to the overflow condition. Generally it is recommended that an attached application should reset the port (ATZ) whenever it receives this response. This overflow condition could occur if (1) too slow of a baud rate is being used, (2) the application is handshaking the RS232 port off for too long of periods of time, or (3) that application has turned on (using the '_I' command) more events than it can handle for the baud rate or application program's speed.

001 = DSS Overload - This response is issued when a 'virtual' DSS is requested (see '_I' Command) and the system has already reached its maximum limit of the number of simultaneous real or 'virtual' DSS instruments that are attached. (NOTE: The maximum limit for *Axxess* Version 2.0 and Version 3.0 is set at 10.)

Example:

Data Queue overflow occurs and data is lost: "ER, 000"

Extension Status Response:

This response gives 'complete' information on an Extension and is when first enabling 'Extension Monitoring' or 'DSS Monitoring' (see '_I' command). Fields at the end of the response are not sent if they haven't changed. This response is only sent if 'Extension Monitoring' or 'DSS Monitoring' Events are enabled.

"EX," <extension>,<status>

The fields of the response are defined as follows:

extension: a digit string that corresponds to the extension being monitored. The string contains from one to five digits and a leading Asterisk (*) if this is the extension number of the attached device.

status: the status (in ASCII) of the extension being monitored:

- '00' = idle
- '01' = busy
- '02' = offline
- '03' = released
- '04' = alerting
- '05' = DND off
- '06' = FWD off
- '07' = DND on
- '08' = FWD on
- '09' = Username

- or -

"EX," <extension>,<status>,<DND MSG|>,<DND text|>

status: the status (in ASCII) of the extension being monitored:
'07' = DND on

DND MSG: this field contains the canned DND text string (top line) that pertains to the monitored extension.

DND text: this field (bottom line) contains the custom text string.

- or -

"EX," <extension>,<status>,<FWD type>,<FWD dest>

status: the status (in ASCII) of the extension being monitored:
'08' = FWD on

FWD type: this field specifies the type of forward mode enabled for the monitored extension.

Types: 0 = FWD Off
1 = FWD Immediate
2 = FWD No Answer
3 = FWD if Busy
4 = FWD if Busy or No Answer

FWD dest: this field specifies the forward destination - username or outside number.

- or -

"EX," <extension>,<status>,<username|>

username: the username associated with the extension being monitored.

Examples:

User Name for Extension 100: "EX, 100, 09, |NORMA|"

Extension 102 just set FWD N/A to JOE B. : "EX, 102, 05, 2, |JOE B.|"

Extension 100 just went into DND: "EX, 100, 07, |OUT TO LUNCH|, |BACK AT 1PM|"

Extension 100 just went out of DND: "EX, 100, 05"

Extension 101 is now busy: "EX, 101, 01"

Status for 'my' Extension (137): "EX, *137, 00"

'Extension Monitoring' - If turned on (using the "_I" command), then the device will begin to receive the status responses for all extensions programmed on 'attached' DSS keys. NOTE: When the Monitoring is first enabled it is possible to receive up to four EX responses per extension (if FWDing and DND are both on).

Example:

Status 'change' for Extension 100: "EX, 100, , 1"

'Hold Call' Response:

This response is issued when the specified call is put on hold at the device. This response is only issued when 'Basic Call Events' are enabled.

"HO," <Call_ID>,<name>, <number >

The fields of the response are defined as follows:

Call_ID: the logical ID assigned to the call.
name: Name of other party - text string - If CO = Caller_ID name, If IC call = username, If unknown it's blank
number: Number of other party: If Outgoing CO = dialed number, If Incoming CO = Caller_ID or ANI number, If IC call = Extension number, Otherwise - blank

Example:

Call @201 is now on Hold on the phone:
"HO,@201,|JANE JONES|, 5551234"

'Info Display' Response:

This response is issued when a special feature/application has requested to use 'Info message displays' to enhance the user interface. This information is equivalent to what is displayed on the top two lines (2 lines by 16 characters) of an LCD keyset. These responses are only sent if the "Info Messages" bit (5) is enabled using the Interrupt Event Command ('_I'):

"ID," <operation>,<field1>, ..., <field 'n'>

The fields of the response are defined as follows:

Operation: 00 - Clear display
01 - Show display
02 - Append display
03 - Show Buffer
04 - Enable Shift Line
05 - Start Timer
06 - Stop Timer
07- Show Timer
08 - Set Timer
09 - Set call cost & time
10 - Show Date and Time
11 - Modify Call Cost Rate

buffer: the display buffer number (0 = static or 1 = transient)
line: line number on display (top line = 0 & bottom line = 1)

offset: character offset (00-15) on <line> where <string> starts
string: Text string to be displayed (wrapping if going beyond end of line).
timer_format: 0 = mm:ss
 1 = hh:mm:ss -- minimum display ":00". Blank filled from left
direction: the direction for counter to count (0=up, 1=down)
timer_value: The initial value of the timer, in seconds.
mask: Line-Clear mask - indicates which lines of display should be cleared before operation occurs: 1 - clear top line, 2 - clear 2nd line, 3 - clear both lines

Examples:

Clear Display: -- Clear lines of buffer as specified by parameters

"ID, 00," <buffer>,<mask>

Show Display: -- Clear lines specified by mask then insert string at specified line/offset

"ID, 01," <buffer>,<line>,<offset>,<mask>,<string>

Append Display: -- Append this string at previous write position

"ID, 02," <buffer>,<string>

Show Buffer: -- Make this buffer active on display.

"ID, 03," <buffer>

Enable Line Shift: -- Enables scroll left of a line (instead of wrapping) when character is written at end of line. Note: Any following ID response (except the *Append Display* response) will disable 'Line Shift' again.

"ID, 04," <line>

Start Timer: -- Start up the timer with the previously set parameters. Doesn't affect whether timer is visible.

"ID, 05"

Stop Timer: -- Stop the timer. Doesn't affect whether timer is visible or not.

"ID, 06"

Show Timer: -- Make timer visible using mask to clear appropriate lines.

"ID, 07," <line>,<mask>

Set Timer: -- Set up initial values of a timer.

"ID, 08,"<timer_format>,<offset>,<direction>,<timer_value>

where:

offset: if = 16 - timer should be centered

Set Call Cost & Time: -- Set up initial values of the call cost counter. Doesn't affect visibility.

"ID, 09,"<timer_value>,<cost>,<rate>

where:

cost: initial cost value in cents

rate: cost rate in cents/minute

Show Date & Time: -- Set up initial values the calendar and show on display line. This display is 'shut off' anytime anything else is written to this line of the display.

"ID, 10," <buffer>,<line>,<mask>,<clock_format>

where:

clock_fomat: 0=12hr, 1=24hr

Change Call Cost Rate: -- Change values of the call cost rate. Doesn't affect visibility.

"ID, 11,"<rate>

Logged Serial Number Response:

This response is always sent in response to a "Log Serial Number" ('_LS') command. The response shows whether the serial number is already in use at another extension number, or if not, that it is in use at the attached extension number.

"LS, " <serial number>,<extension number>

The fields of the response are defined as follows:

<serial number>: Serial number requested. <extension number>: If serial number is in use by another device this is the extension number of that device. If serial number was not in use, it is now considered in use by 'your device' and this is your extension number preceded by a '*' (same response is sent if your port is already using that serial number which allows periodic rechecks of the serial number).

Examples:

The requested Serial Number is already in use on Extension 127: "LS, |A12347|, 127"

The requested Serial Number is now in use on 'your' Extension: "LS, |A12347|, *102"

'Menu Display Response:

These responses allow an application program to make use of the context-sensitive Menu key (sometimes called soft keys) capability provided on an AXXESS system. These responses are issued whenever the 'MENU key region' of the device's display is updated. This Menu Display is to be used in conjunction with the "Menu Key" command described earlier. These responses are only sent if the "Menu Displays" bit is enabled using the Interrupt Event Command ('_I'):

"MD, " <00>,<mask>

- or -

"MD, " <01>,<line>,<offset>,<mask>,<|string|>

The fields of the response are defined as follows:

Operation: 00 - Clear display -- clear requested lines on display
01 - Show display -- clear requested lines on display & display new info

mask: Line-Clear mask - indicates which lines of display should be cleared before operation occurs as follows (in Hexidecimal):

Bit:	7 - msb	6	5	4	3	2	1	0 - lsb
Use:	Reserve	Reserved	Reserved	Reserved	1 - clear 4th line	1 - clear 3rd line	1 - clear 2nd line	1 - clear top line

line: The line of the display (top line = 0 ... bottom line = 3)
 offset: Character offset (0 to 15) on <line> where <string> begins
 string: Text string to be displayed (should wrap to next line if goes beyond position 15)

Assumptions:

1. A menu display is made up of a string of 64 characters (4 lines by 16 characters)
2. Each line is 16 characters, so auto-wrapping should occur at the end of each line.
3. The text on each line can be considered to represent 'keys' - up to 2 keys per line. Odd numbered keys always start in the leftmost column and even-numbered keys always end in the rightmost column.
4. Any time that more than 2 spaces occur in a text string on a line this implies there are 2 'keys'.

Examples:

Clear bottom two lines (3rd and 4th lines) of menu display:

"MD, 00, 0C"

The voice mail system is requesting to clear whatever was previously on the display and then to display two menu keys on the 2nd line to provide listening control:

"MD, 01, 1, 0, 0F, |REWIND FAST FWD|"

'Station Message' Responses:

These responses are issued whenever a station message arrives at, or is removed from, the device AND when 'Station Message Events' are enabled.

Message ON:

"MS,1," <extension>,<name>,<mailbox> - a message has arrived at the device.

Message OFF:

"MS,0," <extension>,<mailbox> - a message has been removed from the device.

The fields of the response are defined as follows:

- extension: a digit string that corresponds to the extension associated with the station message. The string contains from one to five digits.
- name: the username associated with the station message - text string
- mailbox: the mailbox number if the extension is voice mail (only available protocol V01.03 or later).

Example:

"NORMA" at Extension 105 just left a message on the device: "MS, 1, 105, |NORMA|"

Voice mailbox 1201 just left a message on the device: "MS, 1, 299, |Voice Mail Message|, 1201"

Voice mailbox 1201 just canceled a message on the device: "MS, 0, 299, 1201"

'Audible Ring' Response:

This response occurs when the specified call is ringing into the device. This response is only issued when 'Basic Call Events' are enabled.

"RI,"<Call_ID>,<type>,<name1>,<number1>,<volume>,<name2>,<number2>,<name3>,<flags>,<Account>

The fields of the response are defined as follows:

- Call_ID: the logical ID assigned to the call.
- type: the type of the call.

name1: If CO, the 16-character Caller ID name for this call - text string
 If IC, this is the User Name - text string
 number1: If CO, the 10-digit Caller ID number for this call.
 If IC, this is the Extension of the caller
 volume: The volume setting of the ringer (a value of 000 indicates a 'Quiet Ringing'
 call like a Camp-on call, Transfer-to-Hold call, etc.)
 name2: the name of the called party if DID or DNIS - text string
 -or- the name of the trunk if no DID or DNIS - text string
 If IC, this field is blank
 number2: the telephone number of the called party (DID, DNIS)
 If IC, this field is blank
 name3: the username of the device that FWDed or XFRed this call. - text string
 flags: the flags associated with this call.
 Account: the account number (if any) for this call (up to 12 digits)

Example:

A outside (CO) call from "5551212" (his name is unknown) on a call to "ABC Co. at 8002225555" is now ringing directly (not transferred, FWDed, etc.) on the phone:

"RI,@201,01,,5551212,148,|ABC CO.|,8002225555,,0"

'Version Information' Response:

This event is issued in response to a 'Version Request' command ('I').

"VI," <type>,<version>

The fields of the response are defined as follows:

type: 0 = protocol version
 1 = keyset version
 2 = KSU version
 version: If type = 0; <Vxx.yy> where 'xx' = Major version (01 - 99)
 'yy' = Minor version (00 - 99)

The two protocol versions in this document are V01.02 and V01.03 and are available on the following products as of this printing:

V01.02 = *Axxess* Version 2.x Software, *Axxent* Version 1.0
 V01.03 = *Axxess* Version 3.0 Software

If type = 1; <Vxx.yy.d> where 'xx' = Major version (00 - 99)
 'yy' = Minor version (00 - 99)
 'd' = 0 - no LCD, 1 - LCD

If type = 2; <Vxx.yyy> where 'xx' = Major version (00 - 99)
 'yyy' = Minor version (00 - 99)

Example:

In response to an earlier "AT I0" command to check the protocol version, the following event is sent:
 "VI,0,V01.02"

'Waiting for Device' Response:

This response is issued when the specified call is waiting for another device. Some examples are calls that are placed on hold at another device, and calls that camp-on to another device. This response is only issued when 'Basic Call Events' are enabled.

"WT, " <Call_ID>,<type>,<name>, <number>,<volume>

The fields of the response are defined as follows:

Call_ID:	the logical ID assigned to the call.
type:	the type of the call.
name:	Name of other party (text string)
number:	Number of other party
volume:	Volume setting for the 'Busy/MOH' audio.

Example:

The IC Call is to a 'Busy' device is now 'Camped-On':

"WT, @137,0,|NORMA|,100, 128"

Appendix A - AXXESS Feature Codes

KEYSET DEFAULT FEATURE CODES

Outside Trunk Access Codes:

Select Trunk Group 1-99	9201-9299
Automatic Route Selection (ARS)	9200
Emergency Call	911
Outgoing Call	8

Extension Numbers:

Stations	100-211
Hunt Groups	255-274
AxxessLink Applications	275-299
Attendant	0

General Station Feature Codes:

Account Code -- All Following Calls	391
Account Code -- Optional	390
ACD Agent Login	326
ACD Agent Logout	327
ACD Agent Toggle	328
ACD Agent Wrap Up Terminate	329
Answer (Ringing Call)	351
Automatic CO Answer On/Off	360
Automatic Intercom Answer On/Off	361
Automatic Trunk Answer	350
Background Music On/Off	313
Call Forward -- All Calls	355
Call Forward -- If Busy	357
Call Forward -- If No Answer	356
Call Forward -- If No Answer Or Busy	358
Conference	5
Data	340
Default Station	394
Directory	307
Display Time And Date	300
Do-Not-Disturb	370
Do-Not-Disturb Cancel	371
Do-Not-Disturb On/Off	372
Do-Not-Disturb Override	373
Enhanced Speakerphone Enable	310
Feature Key Default	395
Handsfree On/Off	319
Headset On	315
Headset Off	316
Headset On/Off	317
Hold -- Individual	336
Hold -- System	335

Hookflash	330
Hunt Group Remove	322
Hunt Group Replace	323
Hunt Group Remove/Replace	324
Message (Leave a)	365
Cancel Message (That You Left)	366
Cancel Current Message (Your Phone)	368
Silent Message	367
Microphone Mute On/Off	314
Page	7
Page Receive On/Off	325
Program Baud Rate	393
Program Keys	397
Queue (Callback) Request	6
Redial	380
Reminder Message	305
Reminder Message Cancel	306
Reverse Transfer (Call Pick-Up)	4
Review Keys	396
Ring Intercom Always On/Off	377
Ring Tone Selection	398
Station Monitor	321
Station Speed Dial	382
Station Speed Dial Programming	383
Switch Keymap	399
System Forward Enable	352
System Forward Disable	353
System Forward On/Off	354
System Speed Dial	381
Transfer To Hold	346
Transfer To Ring	345

Appendix B - AXXESS Station Key Codes

KEYSET 'Station Key' CODES:

Logical Number	Default Description	Default Value
1:	Dialpad Key 1	Digit 1
2:	Dialpad Key 2	Digit 2
3:	Dialpad Key 3	Digit 3
4:	Dialpad Key 4	Digit 4
5:	Dialpad Key 5	Digit 5
6:	Dialpad Key 6	Digit 6
7:	Dialpad Key 7	Digit 7
8:	Dialpad Key 8	Digit 8
9:	Dialpad Key 9	Digit 9
10:	Dialpad Key 10	Digit *
11:	Dialpad Key 11	Digit 0
12:	Dialpad Key 12	Digit #
13:	User Prog Key 2	
14:	User Prog Key 3	
15:	User Prog Key 4	
16:	User Prog Key 5	
17:	User Prog Key 6	
18:	User Prog Key 7	CNF
19:	User Prog Key 8	HOLD
20:	User Prog Key 9	TRANSFER
21:	User Prog Key 10	SYS SPD
22:	User Prog Key 11	REDIAL
23:	DB Prog Key 1	FWD
24:	DB Prog Key 2	CALL1
25:	DB Prog Key 3	CALL2
26:	DB Prog Key 4	CALL3
27:	DB Prog Key 5	CALL4
28:	DB Prog Key 6	IC
29:	User Prog Key 1	DND
30:	DB Prog Key 8	ANSWER
31:	DB Prog Key 9	OUTGOING
32:	DB Prog Key 10	MUTE
33:	DB Prog Key 11	MSG
34:	DB Prog Key 12	SPKR
35:	DB Prog Key 13	SPCL
36:	DB Prog Key 14	UP
37:	DB Prog Key 15	DOWN
38:	DB Prog Key 16	ACCEPT
39:	DB Prog Key 17	NEXT
40:	DB Prog Key 18	PREV
41:	DB Prog Key 19	BACK
42:	DB Prog Key 20	FORWARD
43:	DB Prog Key 21	CLEAR
44:	DB Prog Key 22	CANCEL

45:	DB Prog Key 23	Menu Key 1
46:	DB Prog Key 24	Menu Key 3
47:	DB Prog Key 25	Menu Key 5
48:	DB Prog Key 26	Menu Key 7
49:	DB Prog Key 27	Menu Key 2
50:	DB Prog Key 28	Menu Key 4
51:	DB Prog Key 29	Menu Key 6
52:	DB Prog Key 30	Menu Key 8

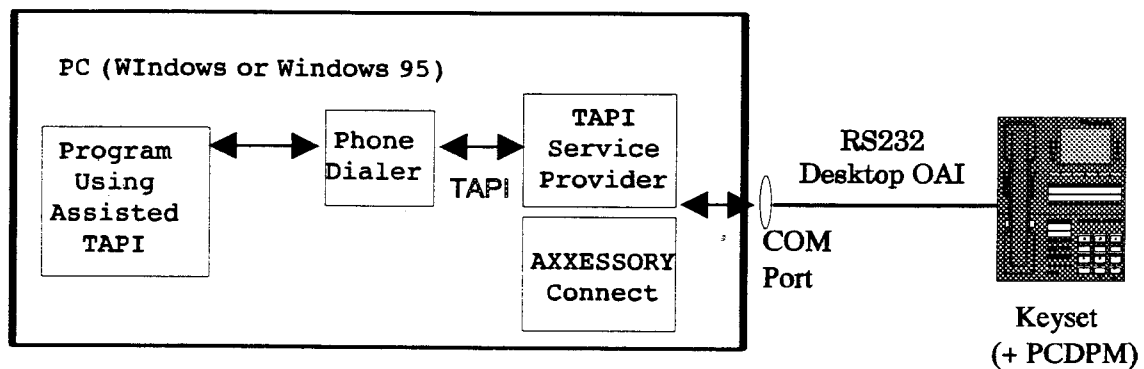
SECTION 4: Application Notes

This section contains numerous short descriptions, usually limited to a few pages each, describing sample applications that could be or have been implemented using the *Axxess Desktop OAI Link*. We welcome descriptions of additional applications that you may come up with that we can share with other *Axxess* OAI developers.

Overview:

This Application Note describes using the **Phone Dialer** which is bundled with Windows 95 (There is a Windows 3.1 version also). **Phone Dialer** is a basic way to have the PC dial the phone number. The program is helpful in testing the installation of the **Inter-Tel TAPI Service Provider** which is bundled with *AXXESSORY Connect* (V2.0 or greater). The program is also used in some cases by other programs (example is Time & Chaos V4.07) to help with making calls which is called Assisted TAPI.

The configuration is as follows:

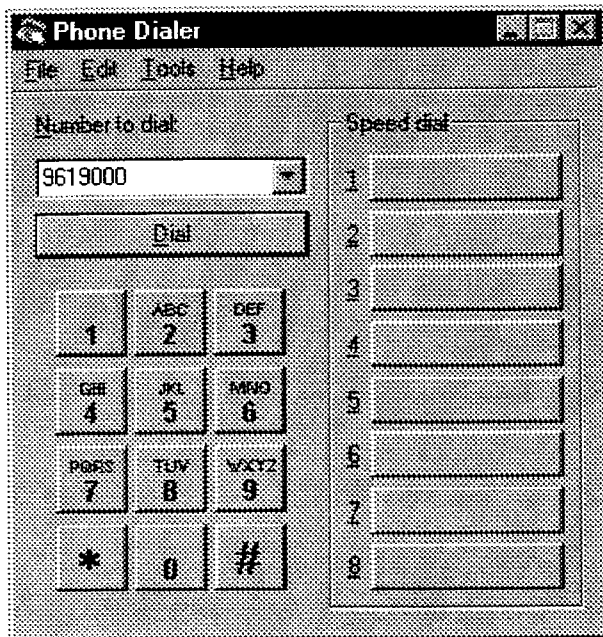


Steps to Setup:

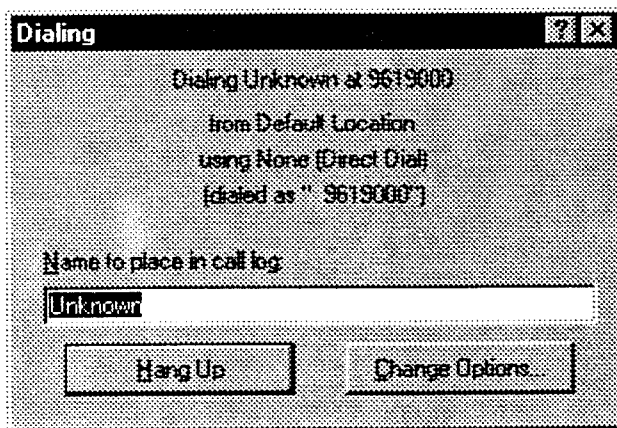
The following shows the steps required for setup:

1. *AXXESSORY Connect* is installed and working correctly. See the *AXXESSORY Connect* Users guide for proper setup information. Note: *AXXESSORY Connect* doesn't have to be active for outbound dialing using **Phone Dialer**.
2. The **Inter-Tel TAPI Service Provider** is installed and working correctly. See the *AXXESSORY Connect* Users guide for proper setup information or the Setting up the TAPI Service Provider Application Note.

The following example shows what the screens look like when **Phone Dialer** is used:



Input the desired number in the Number to dial field and select Dial



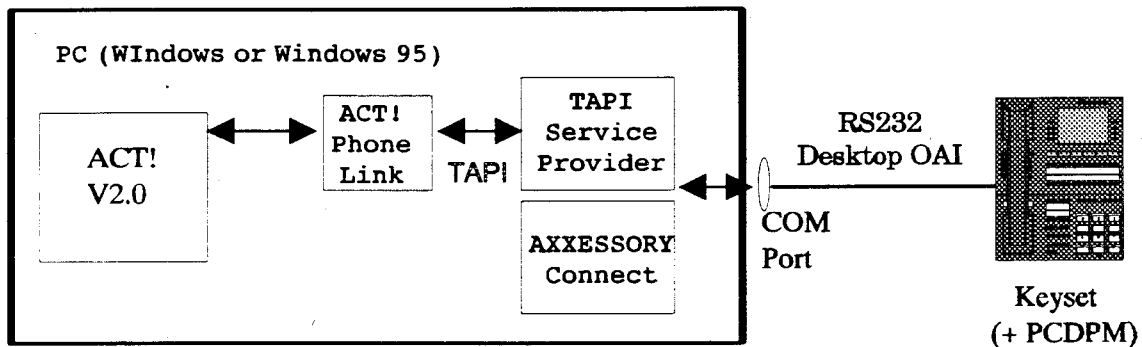
This screen appears as the number is being dialed. Selecting Hang Up will end the call.

Overview:

The popular *ACT! for Windows (V2.04 or greater)* contact management software package, with help from the *ACT! PhoneLink* utility, can tightly integrate with the **Inter-Tel TAPI Service Provider** which is bundled with *AXXESSORY Connect (V2.0 or greater)* software to provide both outdialing from its database and database lookup on incoming calls using TAPI (Not DDE). For outgoing calls, *ACT! PhoneLink* utility intercepts the dial string that *ACT!* thinks it is sending to a modem and redirects it (via a TAPI link) to the **Inter-Tel TAPI Service Provider**. For incoming calls, TAPI messages are sent directly from the **Inter-Tel TAPI Service Provider** to *ACT! PhoneLink* which causes *ACT!* to lookup the phone number in the database and display any matching entries, commonly referred to as "Screen-Pop". The previous method of connecting *ACT! for Windows* and *AXXESSORY Connect* by using DDE and *APPDIAL* is described in the *Linking ACT! for Windows with APPDIAL/AXXESSORY Connect* Application Note.

NOTE: You can not use TAPI and DDE at the same time. If you are upgrading from the DDE method please remove all of the *ACT!* DDE events from *AXXESSORY Connect* and *APPDIAL*.

The configuration is as follows:



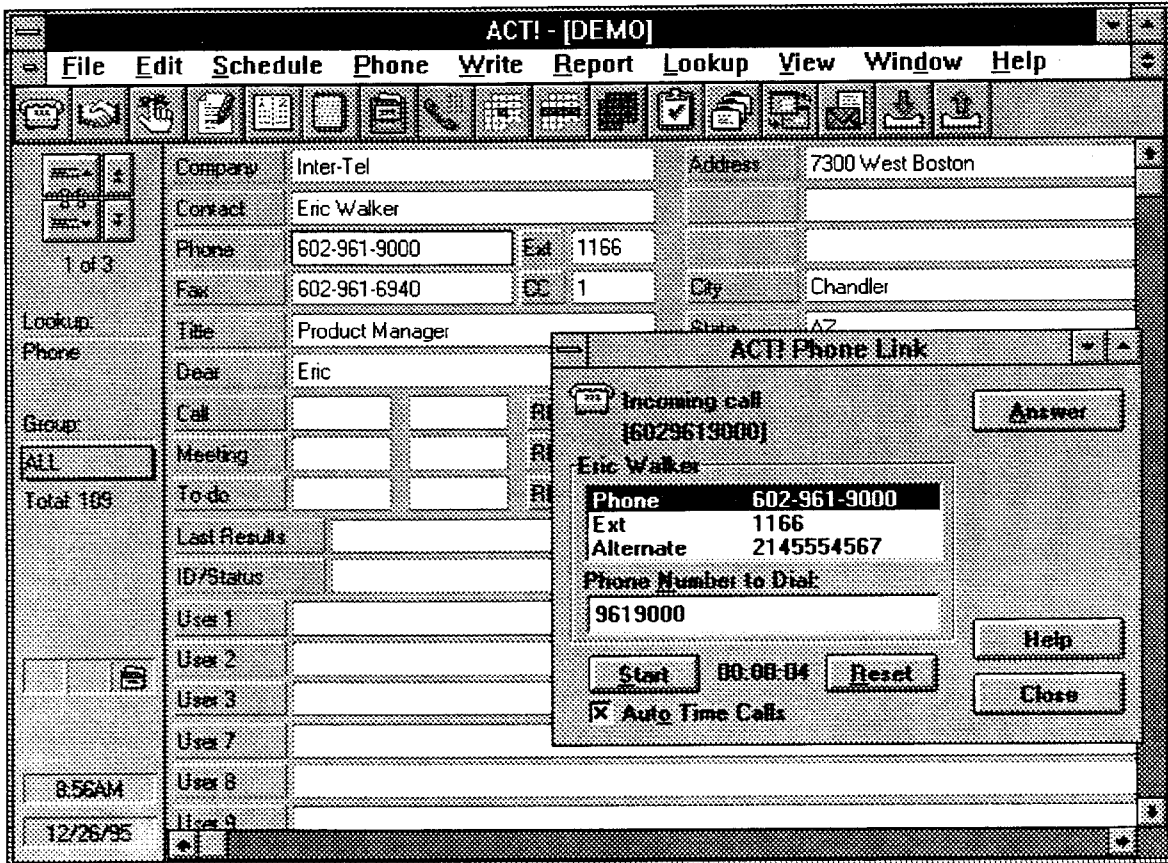
Steps to Setup:

The following shows the steps required for setup:

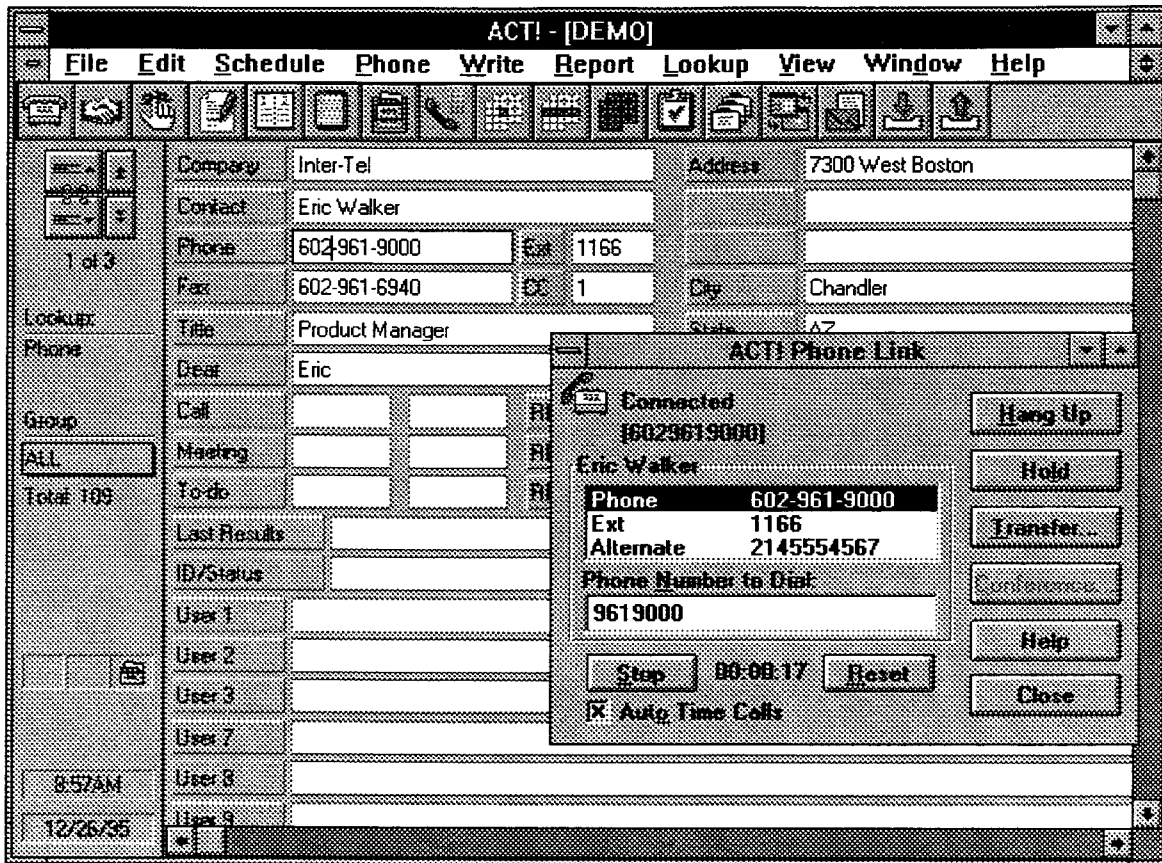
1. *AXXESSORY Connect* is installed and working correctly. See the *AXXESSORY Connect* Users guide for proper setup information. Note: *AXXESSORY Connect* doesn't have to be active for inbound/outbound dialing to *ACT!* when using TAPI.
2. The **Inter-Tel TAPI Service Provider** is installed and working correctly. See the *AXXESSORY Connect* Users guide for proper setup information or the Setting up the TAPI Service Provider Application Note.
3. *ACT!* is installed and working correctly. See *ACT! User's Guide* for proper setup information.
4. *ACT! Phone Link* is installed and working correctly. See *ACT! Phone Link* read me files for proper setup information.

To have your PC ready for inbound/outbound dialing, you need to have **ACT! Phone Link** running, which launches **ACT!**. Do not launch **ACT!** by double-clicking on the **ACT!** icon. **AXXESSORY Connect** does not have to be active, and the **Inter-Tel TAPI Service Provider** is automatically activated by the PC's operating system when needed.

The following example shows the screens that are displayed when a call rings-in, and what options can be done with the call:



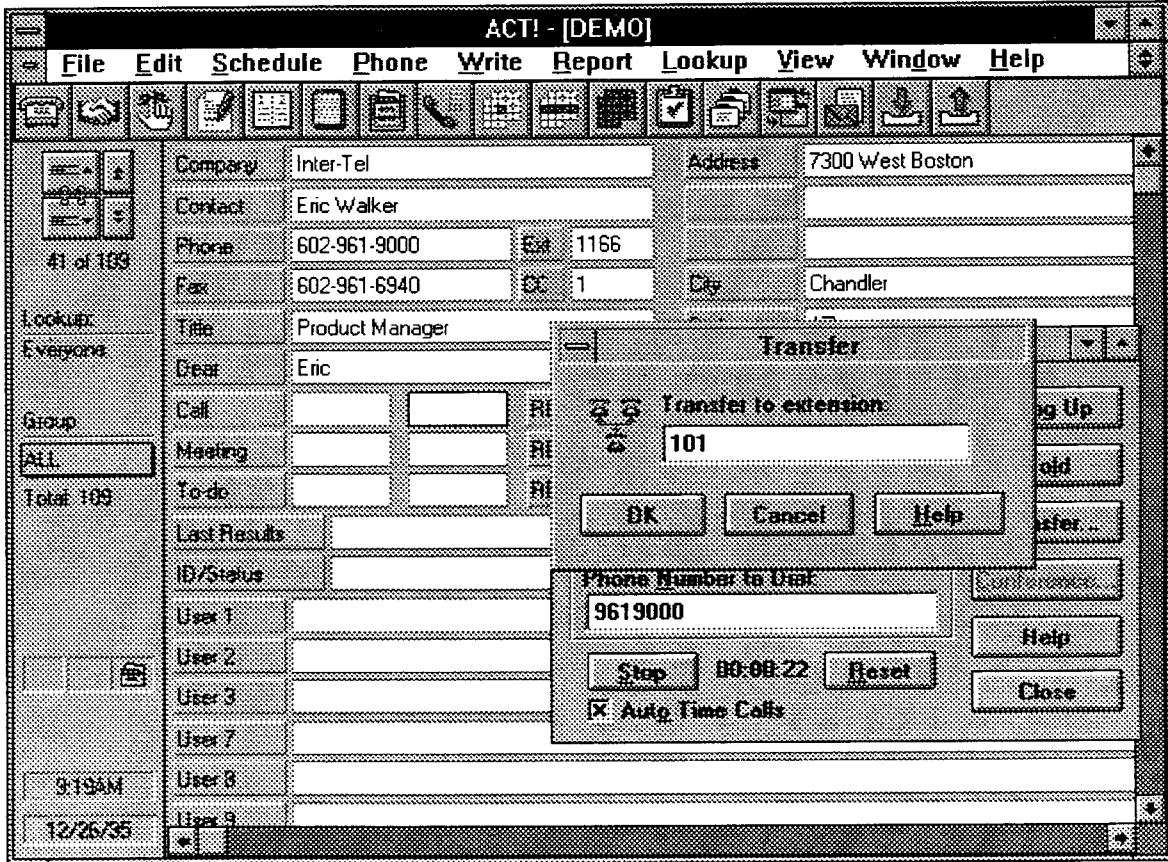
As the call rings-in, the **Inter-Tel TAPI Service Provider** sends the **TAPI** event which causes **ACT! Phone Link** to do a screen-pop with **ACT!**. In this case the database has 3 records matching the Caller-ID phone number, and **ACT!** displays the first matching record.



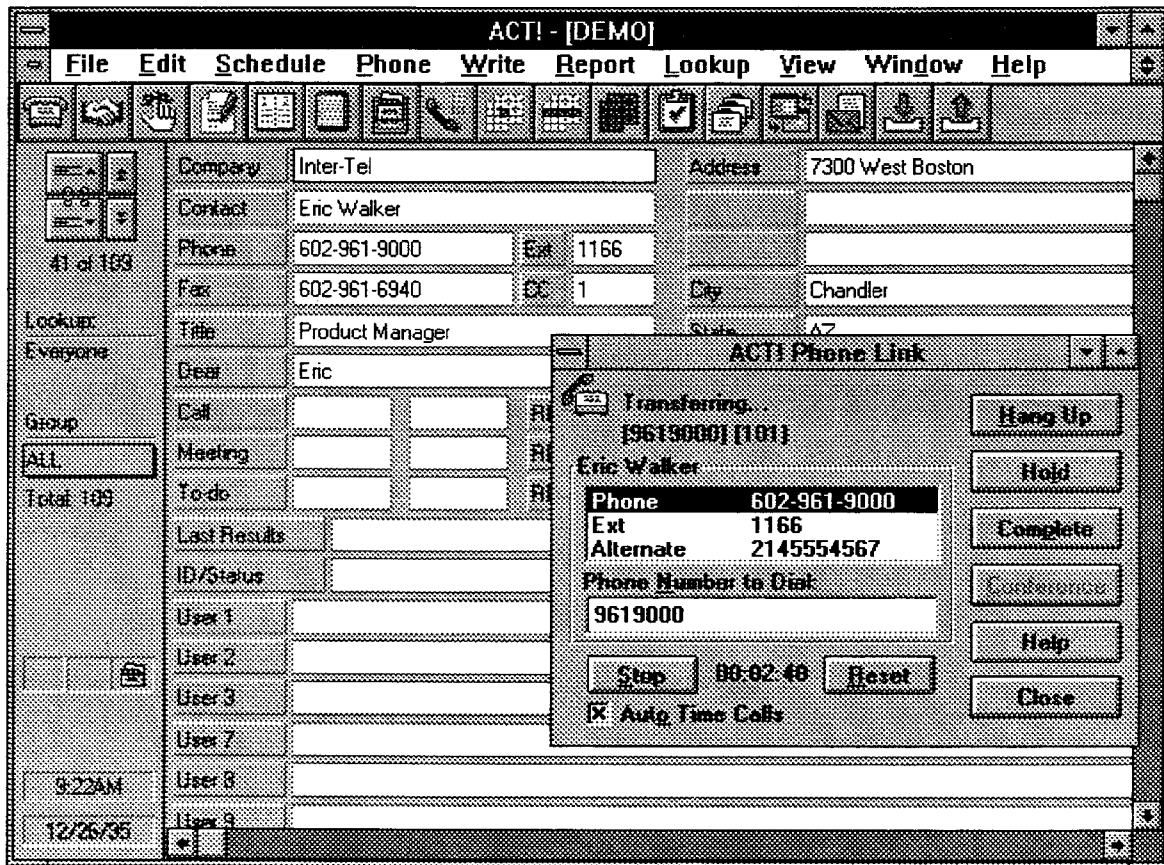
Selecting the Answer button will answer the call and will provide the following options for the call, Hang Up, Hold, Transfer. Note: Conference is not supported with this version of ACT!.

Hang-Up will end the call. Hold will place the call on Hold and the ACT! Phone Link window will have a UnHold button to retrieve the call.

The following screens will show the steps that occur when you select Transfer and want to transfer the call to extension 101:



Selecting OK makes the announcement call to extension 101.



You can announce the call or just select Complete to finish the transfer. Note: The Hang Up and Hold buttons should not be selected. If one of them is selected, the transfer will fail. Also the extension the call is being transferred to must answer the call.

To make a call from *ACT!*, select the desired record, then select the Handset icon from the menu bar. *ACT! Phone Link* will activate, and selecting the Dial button will make the call. Once the call has been started, *ACT! Phone Link* will show the same options described in the call ringing-in example.

Setting up the TAPI Service Provider

10/05/95

Overview:

This application note describes how to setup the Inter-Tel TSPI (Telephony Service Provider) bundled with *Accessory CONNECT* v2.0 for Windows 3.1 & Windows For Workgroups, and Windows 95. The TSPI provides Assisted TAPI, Basic TAPI, and Supplementary TAPI (Hold, Un-Hold, Transfer, Conference) for TAPI compliant applications. We have specific applications notes for integration details for applications we have tested to-date. Please refer any integration questions or additional applications to the Inter-Tel OAI support group. The TSPI works with Inter-Tel AXXENT v1.0 or AXXESS 2.0 (or greater).

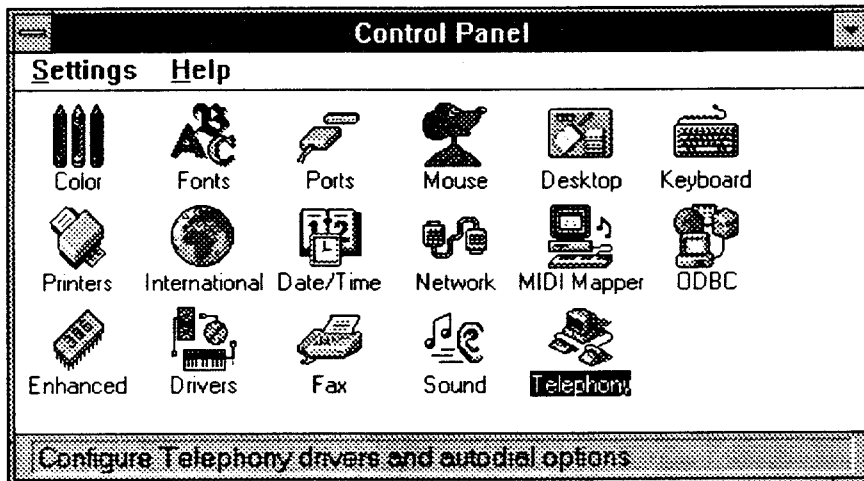
Please note that *Accessory CONNECT* does not have to be active. The TSPI is a separate module that is activated when Windows is started. You may want to have *Accessory CONNECT* active if you want to use it's GUI or for DDE intergration.

Theory of Operation:

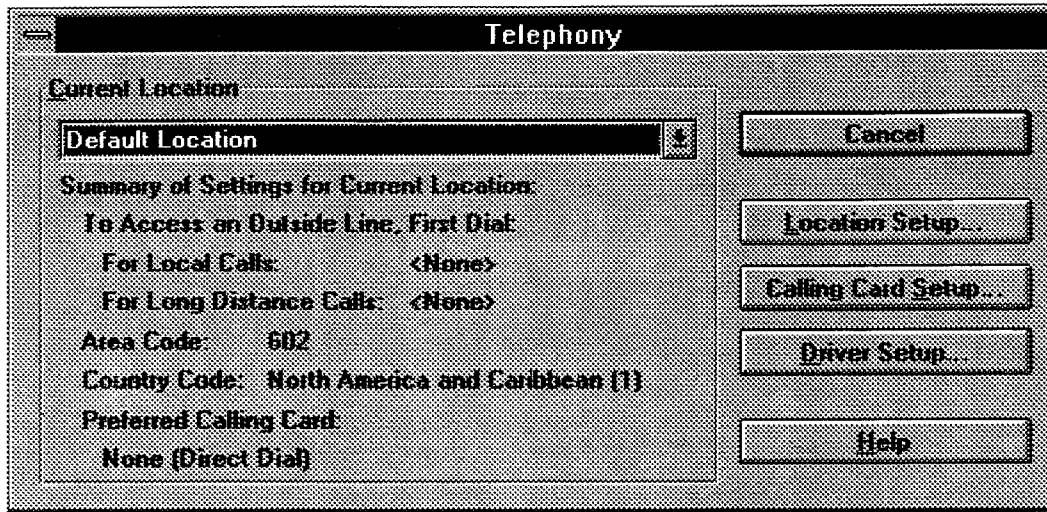
The installation process for *Accessory CONNECT* v2.0 (or greater) adds the Inter-Tel TSPI and the additional TAPI software for Windows 3.1 & Windows for Workgroups. The TAPI software is included in Windows 95. The rest of the application note will detail the setup required.

For Windows 3.1 & Windows for WorkGroups:

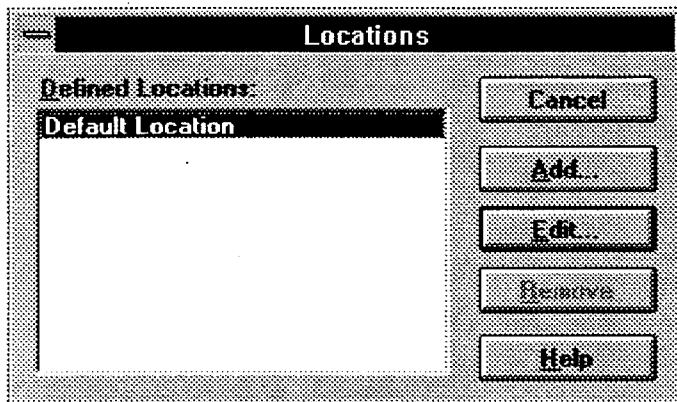
1> Open the Control Panel and select Telephony:



2> Select Location Setup to setup your home area code.



3> Select Edit to modify the location settings.



4> Put your home area code in the Area Code field (example 602), then select OK.

NOTE: Do not put anything in the To Access an Outside Line, First Dial Fields.
It is taken care of in step 7

Edit Location

Location Name:

To Access an Outside Line, First Dial
 For Local Calls For Long Distance Calls

Area Code:

Country Code:

Preferred Calling Card:

Buttons: OK, Cancel, Toll List..., Help, Calling Card Setup...

5> Select Setup to configure the TSPI.

Telephony Drivers

Installed Drivers:

Buttons: Cancel, Add..., Setup..., Remove, Help

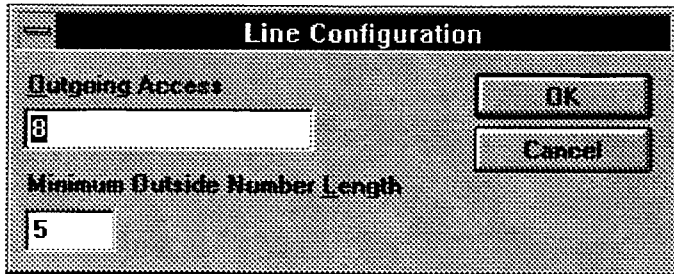
6> Change the Serial Port settings to match the available COM port on the PC.
Then select Configure Line...

Serial Port Configuration

Communications Port:
Baud Rate:
Command Timeout (seconds):

Buttons: OK, Cancel, Configure Line...

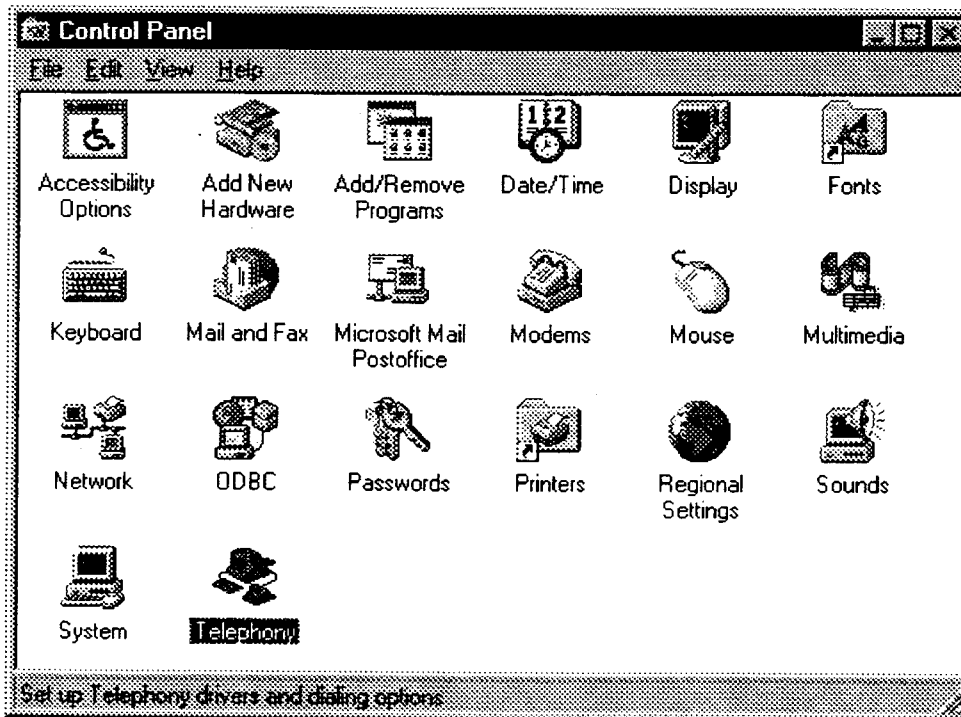
- 7> Change the Outgoing Access field to whatever is required by your phone system to get an outgoing trunk (default is 8). The Minimum Outside Number Length specifies the minimum number of digits required for an outside telephone number. (default is 5).



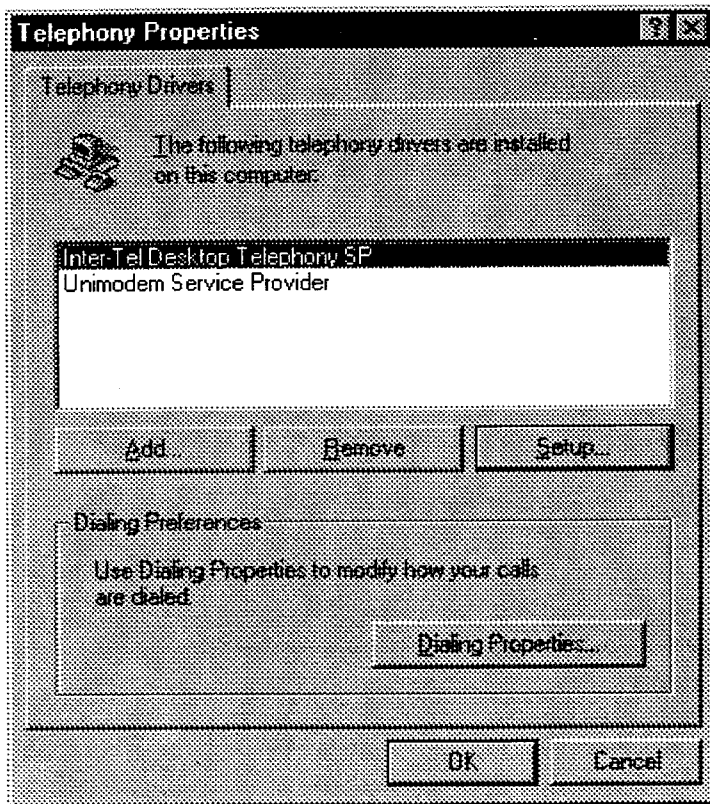
- 8> Everything is setup, close all windows. You may have to restart Windows for the changes to take effect.

For Windows 95:

- 1> Open the Control Panel and select Telephony:

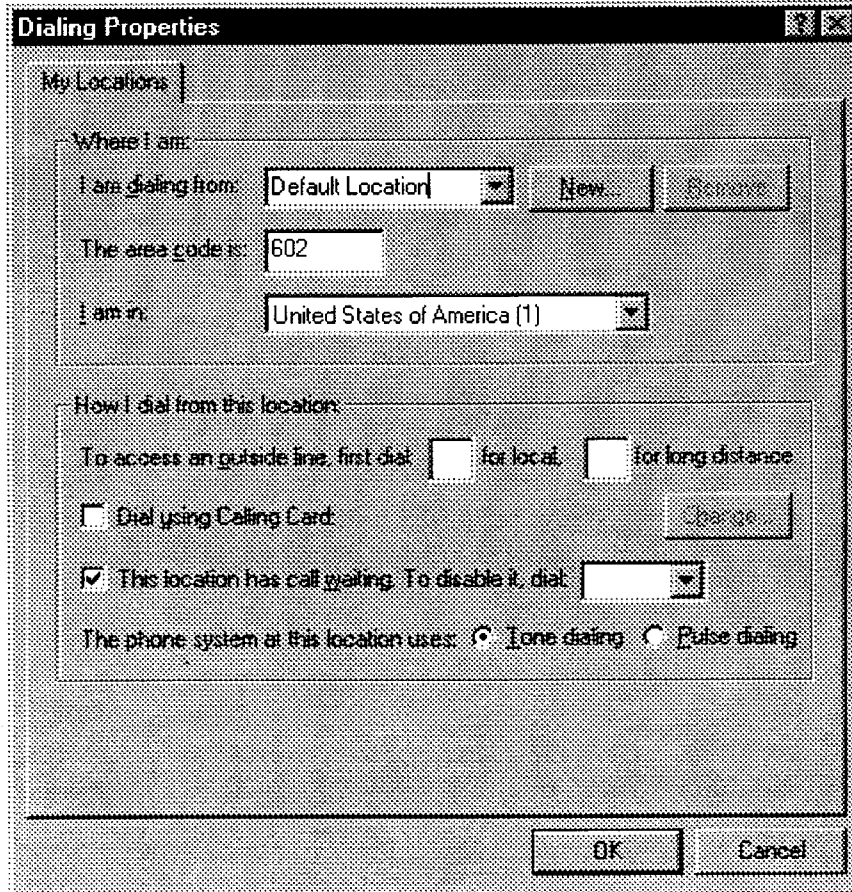


2> Highlight the Inter-Tel Desktop Telephony SP and select Dialing Properties...

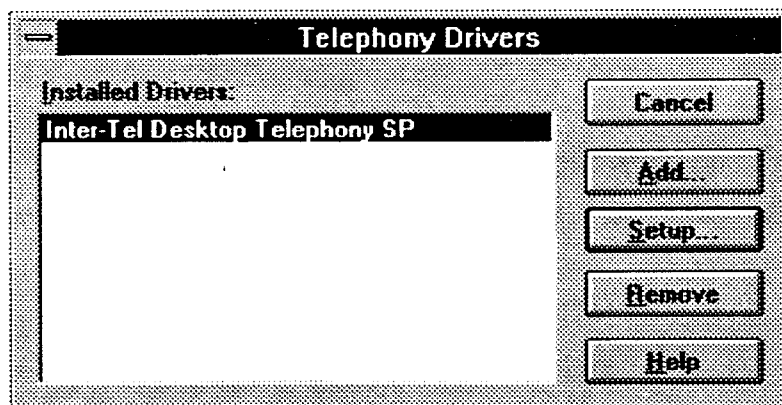


3> Put your home area code in the Area Code field (example 602), then select OK.

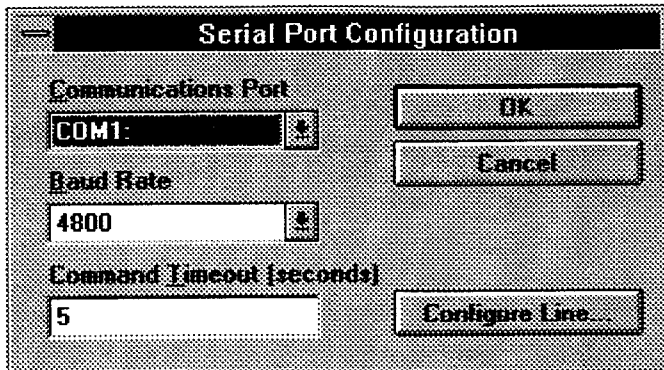
NOTE: Do not put anything in the To Access an Outside Line, First Dial Fields.
It is taken care of in step 7



4> Select Setup to configure the TSPI.

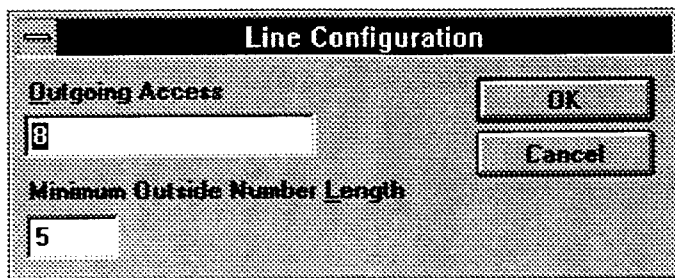


- 5> Change the Serial Port settings to match the available COM port on the PC. Then select **Configure Line...**



The image shows a dialog box titled "Serial Port Configuration". It has three main sections: "Communications Port" with a dropdown menu showing "COM1", "Baud Rate" with a text box containing "4800", and "Command Timeout [seconds]" with a text box containing "5". On the right side, there are three buttons: "OK", "Cancel", and "Configure Line...".

- 6> Change the Outgoing Access field to whatever is required by your phone system to get an outgoing trunk (default is 8). The Minimum Outside Number Length specifies the minimum number of digits required for an outside telephone number. (default is 5).



The image shows a dialog box titled "Line Configuration". It has two main sections: "Outgoing Access" with a text box containing "8", and "Minimum Outside Number Length" with a text box containing "5". On the right side, there are two buttons: "OK" and "Cancel".

- 7> Everything is setup, close all windows. You may have to restart Windows 95 for the changes to take effect.

AXXESS/AXXENT -- Application Note (*AXXESSORY Connect*)

Linking *METZ* Phones with *APPDIAL/AXXESSORY Connect*

10/05/95

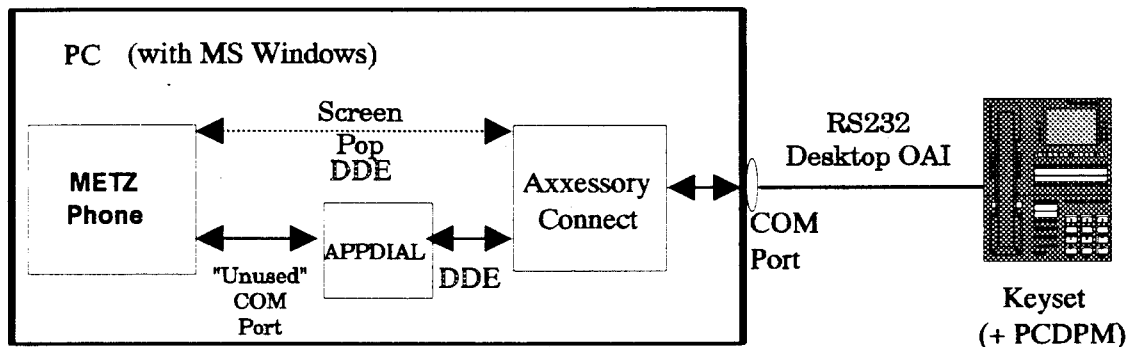
Overview:

The popular *METZ* Phones (V5.51) contact management software package, with help from the *APPDIAL* utility, can tightly integrate with *AXXESSORY Connect* software to provide both outdialing from its database and database lookup on incoming calls. For outgoing calls, *APPDIAL* utility intercepts the dial string that *METZ* Phones thinks it is sending to a modem and redirects it (via a DDE link) to *AXXESSORY Connect*.

NOTE: *APPDIAL* is needed to overcome the limitation of *METZ* Phones not supporting dialing to a DDE device like *AXXESSORY Connect*. It only supports dialing through a modem (or a Hayes-compatible RS232 dialing device). Also the Hangup Button does not work.

For incoming calls, DDE messages are sent directly from *AXXESSORY Connect* to *METZ* Phones to lookup the phone number in the database and display any matching entries, commonly referred to as "Screen-Pop".

The configuration is as follows:



Steps to Setup:

The following shows the setup required to setup *METZ* Phones for Outgoing calls:

1. *AXXESSORY Connect* is installed and working correctly. See the *AXXESSORY Connect* Users guide for proper setup information.
2. *APPDIAL* is installed and working correctly. See the Readme.doc provided with the application for proper setup information. *APPDIAL* should be setup to monitor an "unused" COM port (I. E. It must be defined in the PC's configuration but not currently being used for a modem, mouse, video, etc.). This will be the port used for *METZ* Phones to send the dial information to, and then *APPDIAL* will route the information to *AXXESSORY Connect* to actually dial the number desired.

3. **METZ Phones** is installed and working correctly. See **METZ Phones User's Guide** for proper setup information. Be sure under **Edit, Preferences, Dialing Settings** to set up the phone to dial out of the COM Port being monitored by **APPDIAL**. Also the Windows/Metzpho.ini file needs to be modified to remove the modem initialization string. Here is a portion of the file with the needed changes:

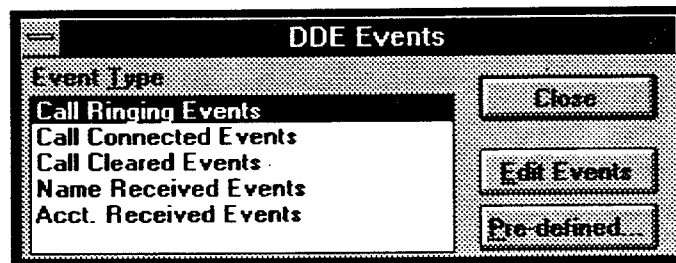
```
[Modem Settings]
PhoModemInit=Q0E&D3 (This should be blank)
Port=2 (Port Number AppDial in monitoring COM #)
```

The following example shows the setup required so when an inbound call is ringing a keyset connected to a PC running **AXXESSORY Connect**:

It will cause **METZ Phones** to do a lookup of the database based on the Caller ID phone number from **AXXESSORY Connect** that matches any of the **PHONE** fields in the **METZ Phones** database.

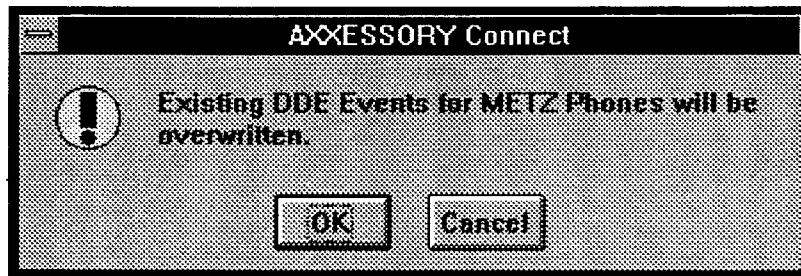
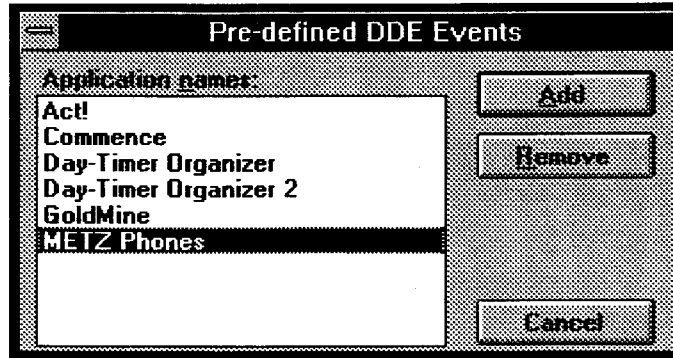
AXXESSORY Connect will have a standard DDE link to support **METZ Phones** (**METZPHO**) for the "Call Ringing" Events (Setup, DDE Events, Call Ringing) DDE Event (**Application Name=METZPhone, Service Name=METZPHO, Topic=METZPHO, Transaction Type=Execute**). The following is a list of the steps required to install the standard DDE link to **METZ Phones**, and how to add your own custom DDE Events if the standard DDE event is not available:

1. Under **Setup** on the main **AXXESSORY Connect** screen select DDE Events:
2. Highlight the desired Event Type and select Edit Events to insert/edit DDE links (See step 4), or select Pre-defined to install the standard **METZ Phones** DDE event.

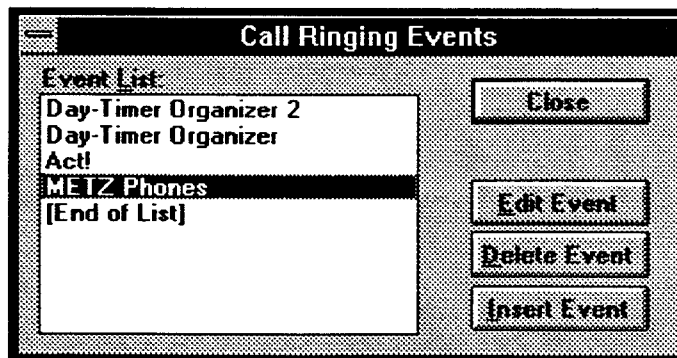


3. For Pre-defined DDE Events, select **METZ Phones** in Application names, and select Add.

Note: If the **METZ Phones** events have already been installed, you will get the error message below and selecting OK will overwrite the existing DDE events. This will remove any custom DDE events with **METZ Phones** as an Application name. If you have created custom DDE events, do not use the same Application names listed in this window (I.E. Use Metz Phones-Custom, etc.).



4. To see what events have been programmed for an event type select the desired event type (Call Ringing Events) and select Edit Events. Then select the desired event from the Event list, and select Edit Event. The DDE events are executed in the order they appear in this list. This is important in cases when you have a chain of events going on between applications to get the desired transaction sequence.



5. The Edit Event window shows the format of the DDE link. Refer to the *Axxessory Connect* manual, (section on DDE Capabilities) for a complete description of the data fields. For a full description of the DDE string sent to *METZ Phones* see their on-line help (Programming with METZ Phones). You may want to modify the FIND command for different parameters.

Edit Call Ringing Event

Application name: METZ Phones

Service Name: METZPHO

Topic: METZPHO

Executable Name: c:\metzpho\metzpho Run if necessary

Transaction Type: Execute Event enabled

DDE Execute / Request / Poke String: [FIND(&P.N.N.Phone Numbers.D.N.Y.N)]

DDE Reply Format:

Audibly Ringing Calls Only Outside calls only

Call Types:

Forward Hold Recall Queue Callback

Transfer Transfer Recall

OK Cancel

Linking a VAX (via KEA! 340) with *AXXESSORY Connect*

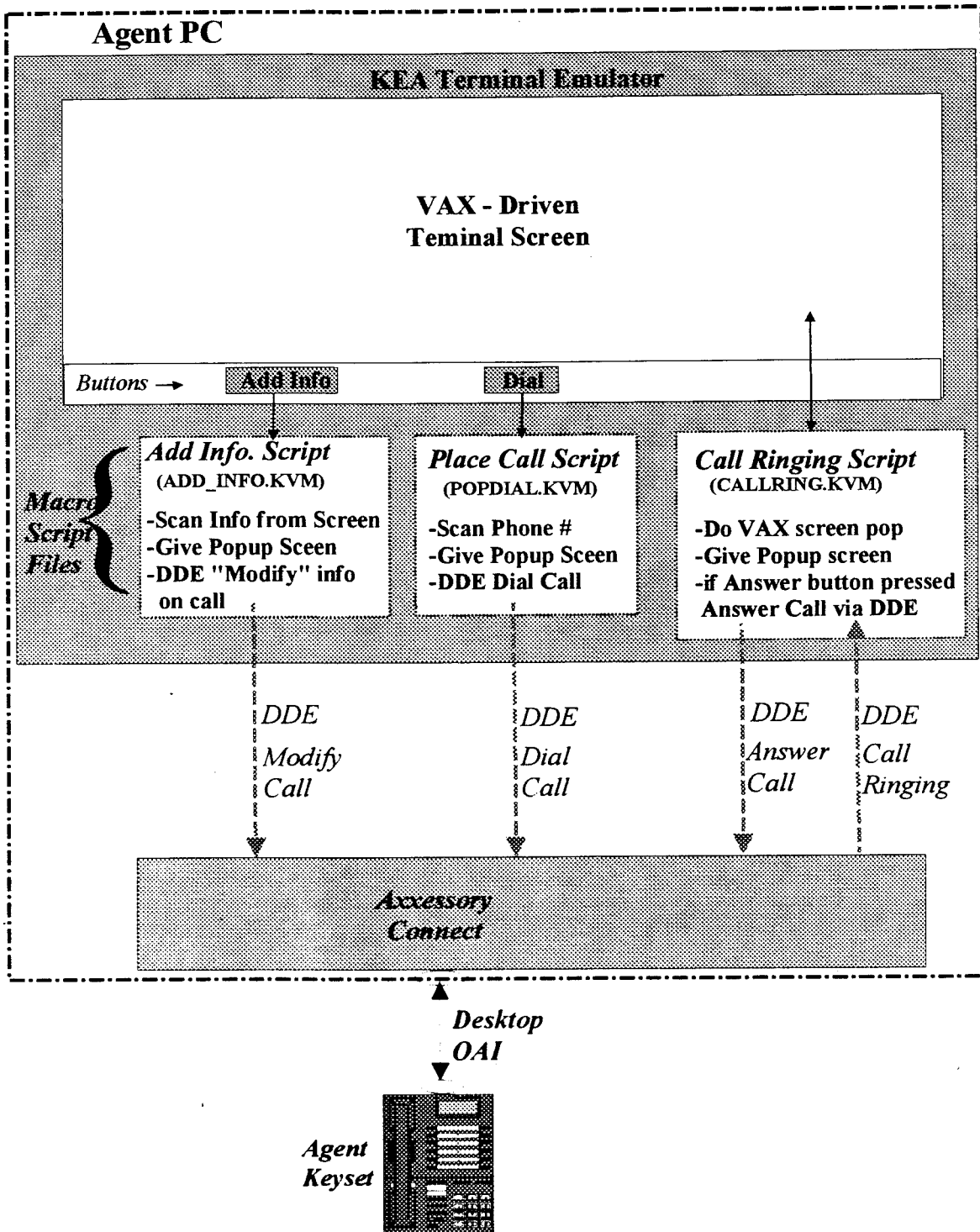
9/6/95

Overview:

A common customer requirement is to provide CTI integration with an existing legacy application running on a VAX 'mainframe'. We've been successful in achieving a very tight CTI integration with just such a VAX application by using *AXXESSORY Connect* (V1.1 or later) and a terminal emulation software package called **KEA! 340** (V4.1) from *Attachmate*. This CTI integration includes: (1) automatically doing a "screen-pop" of the VAX database screen whenever a call arrives with CLID/ANI at the desk, (2) when a call is being handled that arrived without CLID/ANI, the customer information from the VAX screen can be 'scanned' and 'attached' to the call before transferring the call to another desktop thus enabling "screen-pops" to occur with the transferred call, and (3) when placing a call to a customer, with just the click of a "DIAL" button, the customer phone number (and name) can be automatically 'scanned' from the VAX screen, and the call is automatically dialed.

Each 'agent' desktop is equipped with a Windows (3.1) PC to be used instead of a "dumb terminal". This PC is then equipped with *AXXESSORY Connect*, connected to the keyset via a COM port, and the terminal emulation program which attaches to the VAX 'mainframe' using either a COM port or over a LAN. The *AXXESSORY Connect* and the **KEA! 340** programs then communicate with each other using standard DDE commands. The powerful scripting language included with the **KEA! 340** product provided the means to easily design and implement the integration with the customer's VAX program without requiring ANY changes to the legacy VAX applications. (Note: *Attachmate* also has a less expensive **KEA! 420** product that supports the same scripting language features but this was not tested by Inter-Tel at this time.)

The drawing on the following page shows a block diagram of how the "Agent" PC works.



Steps to Setup:

The following example shows the setup required so when an inbound call is ringing a keyset connected to a PC running *AXXESSORY Connect*, it will cause a script macro to be run on the KEA! 340 product to do a lookup on the customer database on the VAX using on the Caller ID phone number from *AXXESSORY Connect*. The following screen shows how to setup to run a script file named "CALLRING" whenever an outside call rings in of the phone.

Macro Script Files:

Following are a two of the sample script files written in the KEA script language. The format of this language is quite similar to "BASIC" so it can be easily read and understood by programmers familiar with that language. This script language requires that any subroutines be placed at the beginning of the file so the 'main' part of the program is always at the end of the file. NOTE: These file listings are intended to be examples of how to use this language and are not guaranteed to work without modification (though they are copied from a actual working installation).

File #1: CALLRING.KTM -- Should be executed whenever a call rings in.

```
//
//
//
//
//
// *****
//
```

```

//      * ParseStr() *
//      *****
//
//      Parse a Comma-Delimited string into fields and return
//      the fields in a list.
//
//      Return:  Count of Fields found
//              0 = Null or blank string was sent
//
Function ParseStr ( CmdStr, Var Fields, MaxFields )
    Var Index      //As Integer
    Var FieldIndx  //As Integer
    Var Endx       //As Integer
    Var FromNa     //As Variant

    Index = 1
    FieldIndx = 1
    LoopDone = False
    While LoopDone = False
        Endx = InStr(CmdStr, ",", 0, Index)
        If Endx = 0
            Endx = Len(CmdStr) //Last field so copy it and exit
            Fields[FieldIndx] = SubStr(CmdStr, Index, Endx - Index + 1)
            LoopDone = True
        Else
            Temp = SubStr(CmdStr, Index, (Endx - Index))
            Fields[FieldIndx] = Temp
            Index = Endx + 1
        End
        FieldIndx = FieldIndx + 1
        If FieldIndx > MaxFields
            LoopDone = True
        End
    End
    Return FieldIndx
End

//
//      *****
//      * PopCall() *
//      *****
//
//      Popup a "Call Received" Dialog Box with an "ANSWER" button.
//
//      Returns:  "I" = Ignore button pressed
//              "A" = Answer button pressed
//              "T" = Timeout occurred
//
Function PopCall (PhoneNo, PhoneName, AcctNo)

Begin Dialog ("Call Received")
    GroupBox ( , True)
        TextBox (" From: " , Width = 20)
        TextBox (" Phone: " , Width = 18)
        TextBox (" Acct: " , Width = 12)
    GroupBreak()

    GroupBox ( , True)
        TextBox ( PhoneName, Width = 20)
        TextBox ( PhoneNo, Width = 18)

```

```

    TextBox ( AcctNo, Width = 12)

GroupBreak()
TextBox ("", Width = 6)
Button ("Answer", Width = 14)
Button ("Ignore", Width = 15, , Offset = 10)
GroupEnd()

Do
    Continue
When Control( "Answer")
    Return "A" //Answer button pressed
When Control ("Ignore")
    Return "I" //Ignore button pressed
When Timeout (250)
    Return "T" //Timeout occurred
End
End

//
//////////////////////////////////// MAIN PROGRAM //////////////////////////////////////
//
// *****
// * New Call Ringing In *
// *****
//
// This Macro should be triggered by a DDE event sent whenever a
// new call rings in. The DDE event should have the following parameter
// format:
// "DDE_Source" "Phone#, Caller_Name, Acct#, Call_ID"
// i.e. "AXXCONNECT" "6029619000, Inter-Tel, 651234, @179"
//
//
Var Fields

If ArgCount > 0 //Place calling program parameters
    //Here if Started Via DDE
    DDEsource = Arg[1] //Copy Arguments
    CmdStr = Arg[2]
Else
    CmdStr = ""
    DDEsource = ""
End

If Len(CmdStr) > 0
    //DDE event received format: "Phone#, Name, Acct#, CallID"
    PCount = ParseStr(CmdStr, Fields, 4 )

    ActivateApp() //Bring us to the foreground
    Restore() //Restore App to normal size

//
// TBD -- Insert code here to automatically initiate a database lookup
// on the attached Host Computer to do a "Screen Pop". This code is
// dependant on the specific 'Legacy' application on the VAX.
//
//Popup a "Calling Ringing" Dialog box
Success = PopCall (Fields[1], Fields[2], Fields[3])
If Success = "A"
    //Answer key was pressed so send DDE to to Answer the call...

```



```

Channel = DDE_Initiate(DDEsource,"SYSTEM")
If Channel > 0
  DDE_Execute (Channel, "[AnswerCall(\"" + Fields[4] + "\")]", 2)
  DDE_Terminate (Channel)
End
End
End
End

```

File #2: POPDIAL.KTM -- Is executed whenever the "DIAL" button is pressed.

```

//
//////////////////////////////////// MAIN PROGRAM //////////////////////////////////////
//
// *****
// * PopDial() *
// *****
//
// This macro pops up a "Place a Call" dialog box allowing the user to
// input/modify a phone number and then dial it. Dialing is accomplished
// by sending a DDE 'Dial' command to Axxessory Connect.
//
//
Var PhoneNumber
Var PhoneName
Var Account

//Initialize all params to blank
PhoneNumber = ""
PhoneName = ""
Account = ""

//
// TBD -- Copy 'PhoneNumber', 'Name', and 'Account' number from the
// appropriate position on the screen.
//
//
//If No phonenumber available then popup a Dialog box..
If Len (PhoneNumber) = 0
  Begin Dialog ("Place a Call")
    GroupBox ( , True)
      EditText ("Phone:", PhoneNumber, Width = 25)
      EditText ("Name:", PhoneName, Width = 25)
    GroupBreak()
      EditText ("Account: ", Account, Width = 15)
    GroupBreak()

      TextBox ("", Width = 6)
      Button ("Dial", Width = 14)
      Button ("Cancel", Width = 15, , Offset = 10)
    GroupEnd()

  Do
    Continue
    When Control( "Dial") //Dial button pressed so continue..

    When Control ("Cancel")
      Return
    When Timeout (750)
      Return
  End

```

```

End

If Len (PhoneNumber) > 0
  If Len(PhoneNumber) > 5 //Outside call
    DialStr = "[DialNumber(\\" + PhoneName +
    "\", \\" + Account + "\")]"
  Else
    If Len(PhoneNumber) > 0 //Intercom Call
      DialStr = "[DialNumber(\\" + PhoneNumber + "\")]"
    Else
      Return
    End
  End
End

//Start DDE conversation
Channel = DDE_Initiate("AXXCONNECT", "SYSTEM",)
If Channel > 0
  //Send DDE Dial command
  DDE_Execute (Channel, DialStr, 2 )
  DDE_Terminate (Channel)
End

End

```

Linking SelectPHONE CD with AXXESS (Desktop OAI)

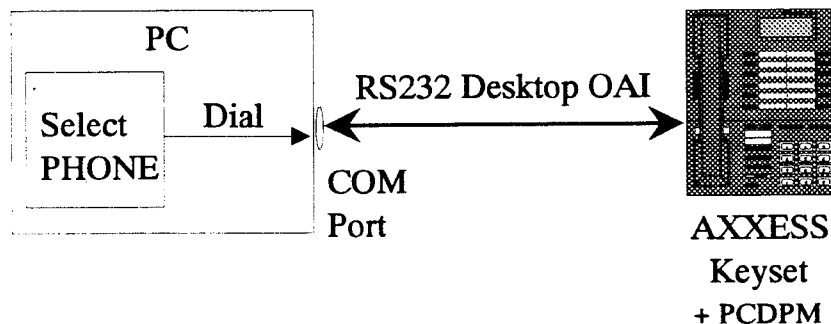
5/23/95

Overview:

A new group of programs, called **CD PhoneBooks**, are now available for PCs. These CD Phonebooks,™ use CD ROMs to store huge databases containing the equivalent of many entire phone books that can be easily searched and sorted using a standard PC. Some of these programs then allow you to select an entry from these databases and press a key to 'speed dial' this person/business. Thus, the user has the equivalent of over 80 million 'speed dial' numbers, complete with names and addresses.

We tested one of these programs, called **SelectPHONE CD 1995** (made by **ProPhone**), with the *Axxess* system and Desktop OAI. **SelectPHONE CD** dials using a "Hayes-compatible" command, and since the *AXXESS* desktop OAI link directly supports Hayes-compatible commands, the two worked together easily. This direct connection to an *Axxess* keyset Desktop OAI port eliminates the need for a modem and an additional analog phone line on the desktop. This application note describes the steps required to setup the Desktop OAI Interface in the *AXXESS* System and the setup required for the application program to do out-bound dialing.

Following is drawing showing the connection:



Steps to Setup:

A. On the AXXESS system: (Note: See the *AXXESS Installation and Field Maintenance Manual* for complete details on the setup procedure.)

1. The *AXXESS* System must be running version 2.0 software or greater with the Desktop OAI Interface premium feature enabled.
2. A PCDPM must be installed in the Keyset (Phone).
3. The station circuit for the Keyset must be configured for use with a 'Desktop Interface'.
4. From the Keyset, enter the Program Baud Rate (393) Feature Code to change to 1200 baud. Note: **SelectPHONE CD** doesn't seem to have the ability to change baud rates (1200 baud only).

B. On the PC:

1. Install the **SelectPHONE CD (ProPhone)** software on the PC.
2. Attach the PCDPM RS232 cable to the COM port of the PC.
3. In **ProPhone** use **File, Dial Setup** and select the correct COM port.
4. Search the database, select an entry, and press the Dial Icon to automatically dial the number. The *Axxess* keyset will then use the speakerphone to automatically make the call.
5. When the call is completed simply hangup using the keyset.

Note: It is NOT presently possible to use the *AppDial* widget software with **SelectPHONE CD**.

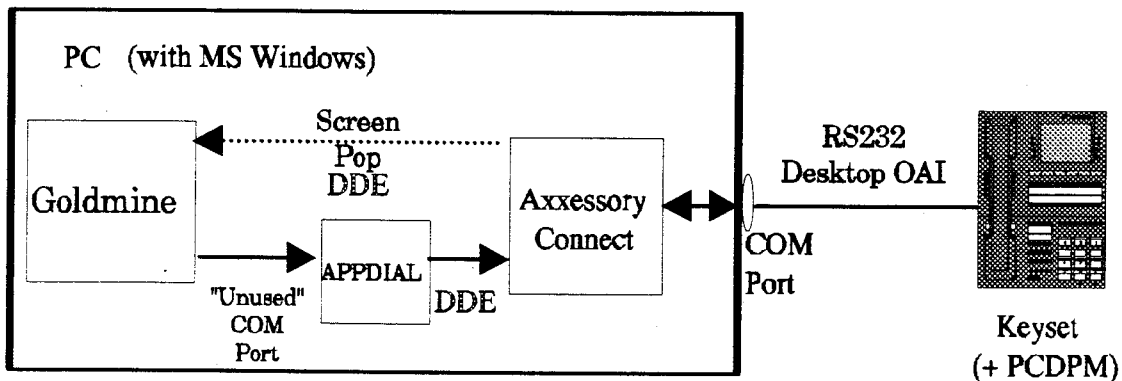
AXCESS – Application Note (AXCESSORY Connect)

Linking GoldMine for Windows with APPDIAL/AXCESSORY Connect

2/17/95

Overview:

The popular *GoldMine for Windows (V2.5a)* contact management software package, with help from the *APPDIAL* utility, can tightly integrate with *AXCESSORY Connect* software to provide both outdialing from its database and database lookup on incoming calls. For outgoing calls, *APPDIAL* utility intercepts the dial string that *GoldMine* thinks it is sending to a modem and redirects it (via a DDE link) to *AXCESSORY Connect*. **NOTE: *APPDIAL* is needed to overcome the limitation of *GoldMine* not supporting dialing to a DDE device like *AXCESSORY Connect*. It only supports dialing through a modem (or a Hayes-compatible RS232 dialing device).** For incoming calls, DDE messages are sent directly from *AXCESSORY Connect* to *GoldMine* to lookup the phone number in the database and display any matching entries, commonly referred to as "Screen-Pop". The configuration is as follows:



Steps to Setup:

The following shows the setup required to setup *GoldMine* for Outgoing calls:

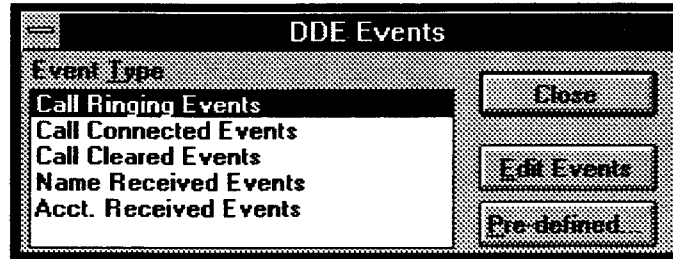
1. *AXCESSORY Connect* is installed and working correctly. See the *AXCESSORY Connect* Users guide for proper setup information.
2. *APPDIAL* is installed and working correctly. See the Readme.doc provided with the application for proper setup information. *APPDIAL* should be setup to monitor an "unused" COM port (I. E. It must be defined in the PC's configuration but not currently being used for a modem, mouse, video, etc.). This will be the port used for *GoldMine* to send the dial information to, and then *APPDIAL* will route the information to *AXCESSORY Connect* to actually dial the number desired.
3. *GoldMine* is installed and working correctly. See *GoldMine* User's Guide for proper setup information. Be sure under **Edit, Preferences, Modem Setup** to set up the phone to dial out of the COM Port being monitored by *APPDIAL*.

The following example shows the setup required so when an inbound call is ringing a keyset connected to a PC running *AXCESSORY Connect*:

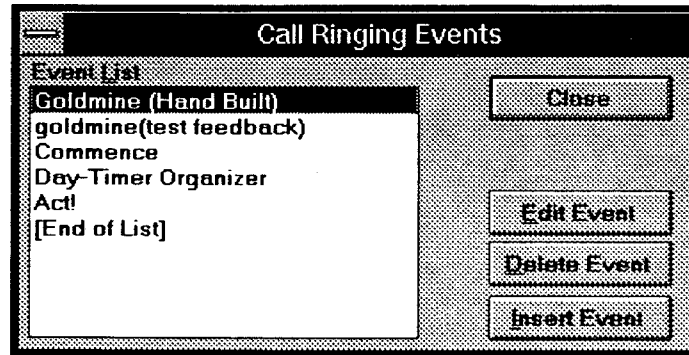
It will cause *GoldMine* to do a lookup of the database based on the Caller ID phone number from *AXCESSORY Connect* that matches the PHONE1 field in the *GoldMine* database.

The following is a list of the steps required to install the *AXXESSORY Connect* DDE link to *GoldMine*, and how to add your own custom DDE Events:

1. Under **Setup** on the main *AXXESSORY Connect* screen select DDE Events:
2. Highlight the desired Event Type (Call Ringing Events) and select Edit Events to insert/edit DDE links.



3. To see what events have been programmed for an event type select the desired event type (Call Ringing Events) and select Edit Events. Then select Insert Event to add a new event or select Edit Event. The DDE events are executed in the order they appear in this list. This is important in cases when you have a chain of events going on between applications to get the desired transaction sequence.



4. The Edit Event window shows the format of the DDE link. Refer to the *Axxessory Connect* manual, (section on DDE Capabilities) for a complete description of the data fields.

You may want to adjust the parameters in the DDE Execute String which modifies how *GoldMine* responds to the *Axxessory Connect* DDE event when a call rings in:

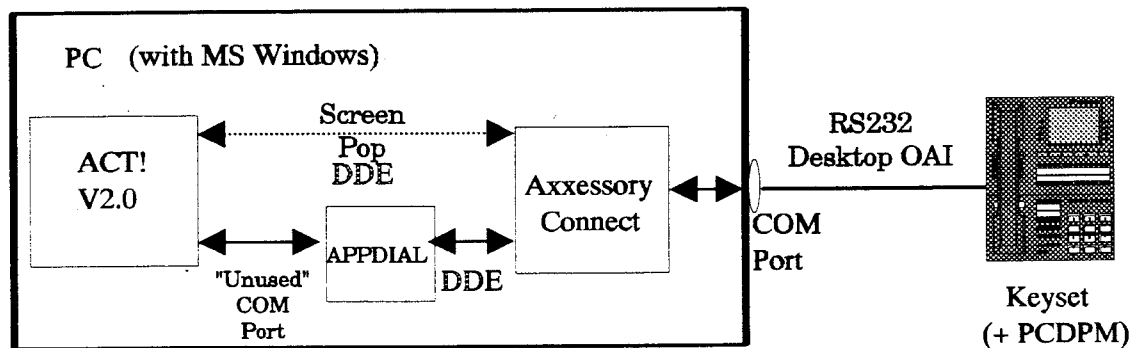
[CALLERID ("&P", "&N", X)] X can be 0, 1, or 2 (The default is 0)

See the *GoldMine* Manual for further details about this DDE Command.

- 0 Puts up a window box that gives you the following options:
 Open: Does a search and shows match in a new record object.
 Goto: Does a search and shows match in the current record object.
 Ignore: Leaves GoldMine unchanged.
- 1 Does a search and shows match in a new record object.
- 2 Does a search and shows match in the current record object.

Overview:

The popular *ACT! for Windows (V2.0)* contact management software package, with help from the *APPDIAL* utility, can tightly integrate with *AXXESSORY Connect* software to provide both outdialing from it database and database lookup on Incoming calls. For outgoing calls, *APPDIAL* utility intercepts the dial string that *ACT!* thinks it is sending to a modem and redirects it (via a DDE link) to *AXXESSORY Connect*. **NOTE: *APPDIAL* is needed to overcome the limitation of *ACT!* not support dialing to a DDE device like *AXXESSORY Connect*. It only supports dialing through a modem (or a Hayes-compatible RS232 dialing device).** For incoming calls, DDE messages are sent directly from *AXXESSORY Connect* to *ACT!* to lookup the phone number in the database and display any matching entries, commonly referred to as “Screen-Pop”. The configuration is as follows:



Steps to Setup:

The following shows the setup required to setup *ACT!* for Outgoing calls:

1. *AXXESSORY Connect* is installed and working correctly. See the *AXXESSORY Connect* Users guide for proper setup information.
2. *APPDIAL* is installed and working correctly. See the Readme.doc provided with the application for proper setup information. *APPDIAL* should be setup to monitor an “unused” COM port (I. E. It must be defined in the PC’s configuration but not currently being used for a modem, mouse, video, etc.). This will be the port used for *ACT!* to send the dial information to, and then *APPDIAL* will route the information to *AXXESSORY Connect* to actually dial the number desired.
3. *ACT!* is installed and working correctly. See *ACT!* User’s Guide for proper setup information. Be sure under Edit, Preferences, Dialing Settings to set up the phone to dial out of the COM Port being monitored by *APPDIAL*.

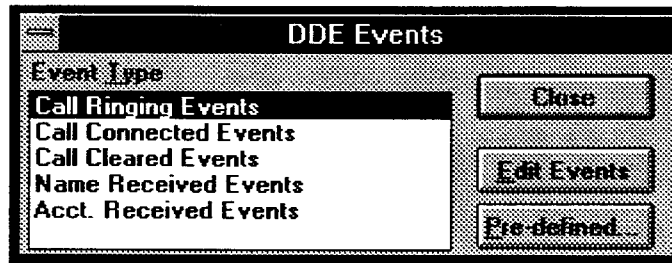
The following example shows the setup required so when an inbound call is ringing a keyset connected to a PC running *AXXESSORY Connect*:

It will cause *ACT!* to do a lookup of the database based on the Caller ID phone number from *AXXESSORY Connect* that matches the PHONE field in the *ACT!* database. Note that *ACT!* uses hyphenated phone numbers in its PHONE field, so *AXXESSORY Connect* DDE event must be

configured to send a hyphenated phone number (&H). The format for (&H) is ####-###-#### where '#' is a valid number and '-' is a required delimiter for *ACT! for Windows (V2.0)*.

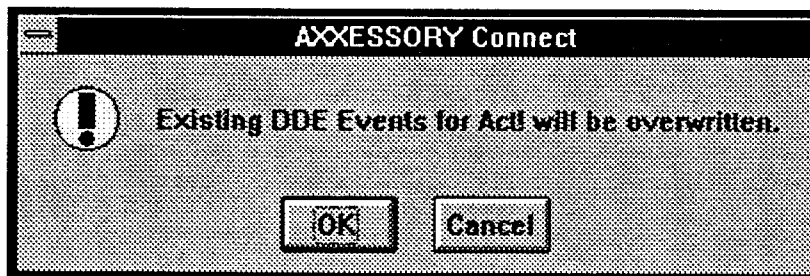
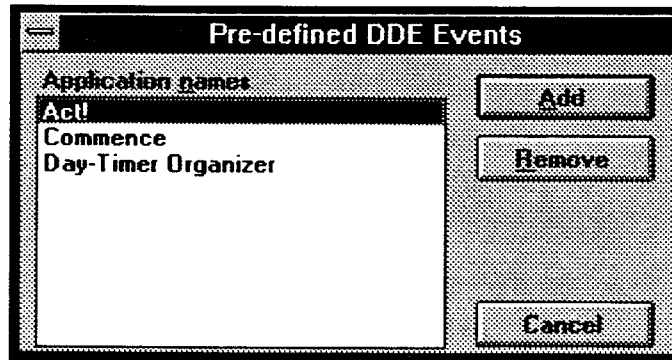
AXXESSORY Connect has a standard DDE link to support *ACT!* (ACTWIN2) for the "Call Ringing" Events (Setup, DDE Events, Call Ringing) DDE Event (Application Name=Act!, Service Name=ACTWIN2, Topic=SYSTEM, Transaction Type=Execute). The following is a list of the steps required to install the standard DDE link to *ACT!*, and how to add your own custom DDE Events:

1. Under **Setup** on the main *AXXESSORY Connect* screen select DDE Events:
2. Highlight the desired Event Type and select Edit Events to insert/edit DDE links (See step 4), or select Pre-defined to install the standard *Act!* DDE event.

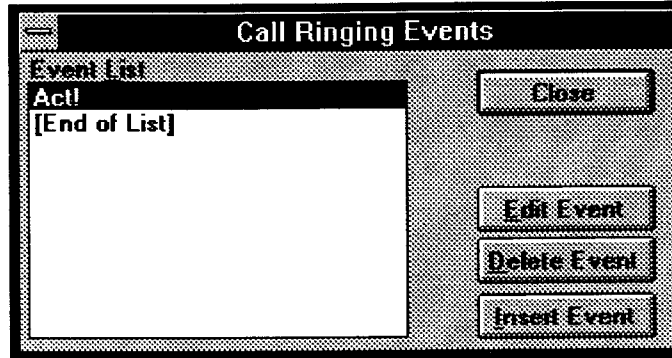


3. For Pre-defined DDE Events, select *Act!* in Application names, and select Add.

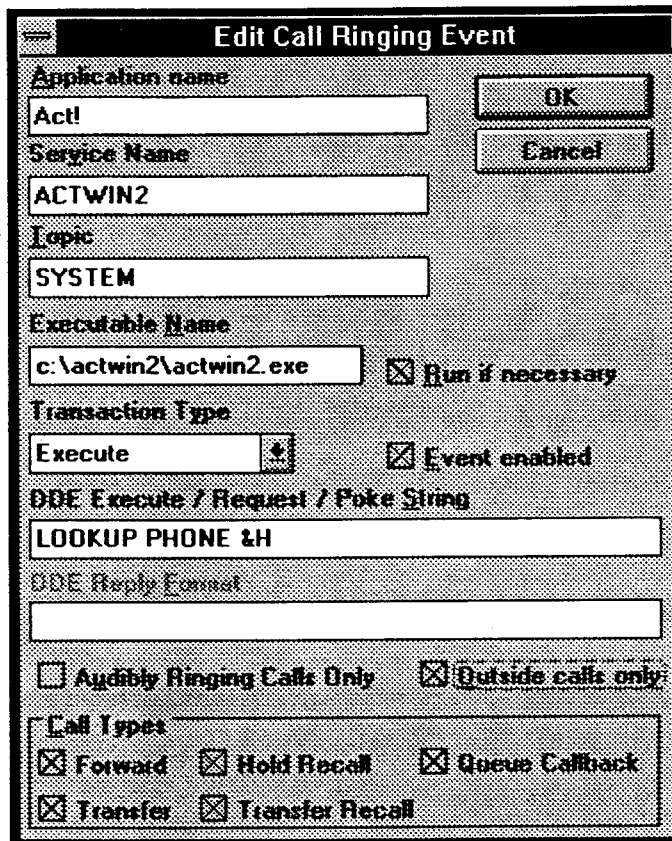
Note: If the *Act!* events have already been installed, you will get the error message below and selecting OK will overwrite the existing DDE events. This will remove any custom DDE events with *Act!* as an Application name. If you have created custom DDE events, do not use the same Application names listed in this window (I.E. Use Act!-Custom, etc.).



4. To see what events have been programmed for an event type select the desired event type (Call Ringing Events) and select Edit Events. Then select the desired event from the Event list, and select Edit Event. The DDE events are executed in the order they appear in this list. This is important in cases when you have a chain of events going on between applications to get the desired transaction sequence.



5. The Edit Event window shows the format of the DDE link. Refer to the *Accessory Connect* manual, (section on DDE Capabilities) for a complete description of the data fields.



Linking *COMMENCE* 2.0 or 2.1 with *AXXESSORY Connect*

12/14/94

Overview:

The "*Commence*" contact management software package (V2.0 or 2.1) can be easily integrated with the *AXXESSORY Connect* software to provide an integrated solution for both (1) dialing calls out directly from the *Commence* database and (2) providing database lookup and 'Screen-Pop' for incoming calls ringing in with CallerID, ANI, or DNIS information.

Theory of Operation:

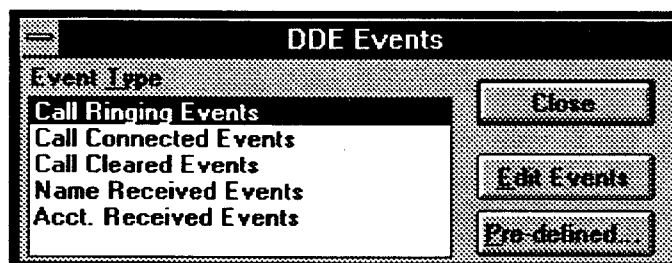
Commence is setup by the user to contain an address book "database" of "contacts" (people/companies that they deal with). This database contains the names, addresses, phone numbers and other pertinent information of each contact. When placing an outgoing call to a contact, the user can simply lookup the contact in the *Commence* address book and press the 'phone' menu icon, which automatically activates *AXXESSORY Connect* to place the call through the phone system, and provide the contact "name" to the phone system for use on display phones and system logging.

Conversely, on incoming calls, *AXXESSORY Connect* provides *Commence* with phone call information from the phone system. This is accomplished by *AXXESSORY Connect* requesting a match on the 'Caller ID' number and *Commence* will return the name from the first record it finds in its database that matches the 'Caller ID' exactly. It will try to match on: Business Phone, Fax Number, Home Phone. NOTE: The phone fields you are trying to match on must be visible in the *Commence* Address Book. If there are no matches the call will be ignored by *Commence* (I.E. The call will keep ringing). If *Commence* returns a name, then the Name Received DDE Event will send a command to *Commence* to do a Screen-Pop 'Database lookup' based on the Name and 'Caller ID' specified. From *Commence* the user is then provided with choices of: (1) **OK** (Closes the message window but does not answer the call), (2) **Answer** (Answers the call in a hands-free mode and displays the database record that matched the call), (3) **Send To V-Mail** (Avoids the caller by Transferring the call to Voice Mail), (4) **Snooze** (You enter the # of minutes to wait before displaying this message box and does not change the status of the call). Note: If the user answers the call using the keyset keys or handset, they should NOT also use the **Answer** from message box because this will cause *AXXESSORY Connect* to see another 'Offhook' request which will terminate the call.

Steps to Setup:

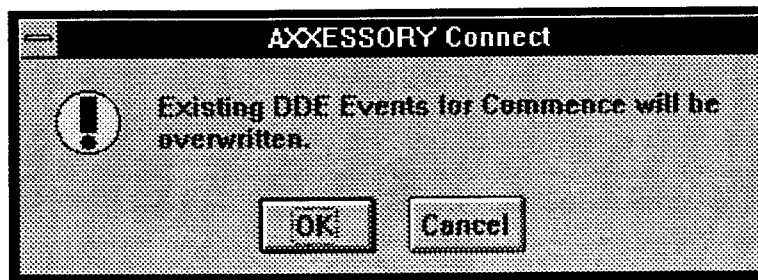
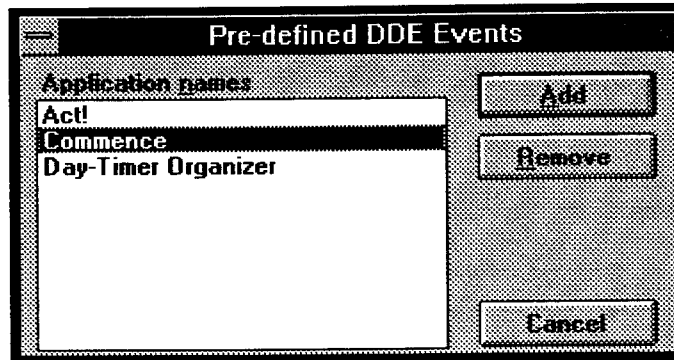
AXXESSORY Connect has a standard DDE links to support *Commence* for the "Call Ringing" and "Name Received" Events (Setup, DDE Events). The following is a list of the steps required to install the standard DDE link(s), and how to add your own custom DDE Events:

1. Under Setup on the main *AXXESSORY Connect* screen select DDE Events:
2. Highlight the desired Event Type and select Edit Events to insert/edit DDE links (See step 4), or select Pre-defined to install the standard *Commence* DDE event(s).

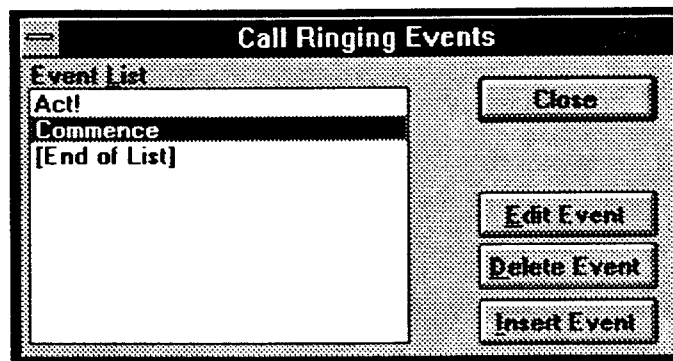


3. For Pre-defined DDE Events, select *Commence* in Application names, and select Add.

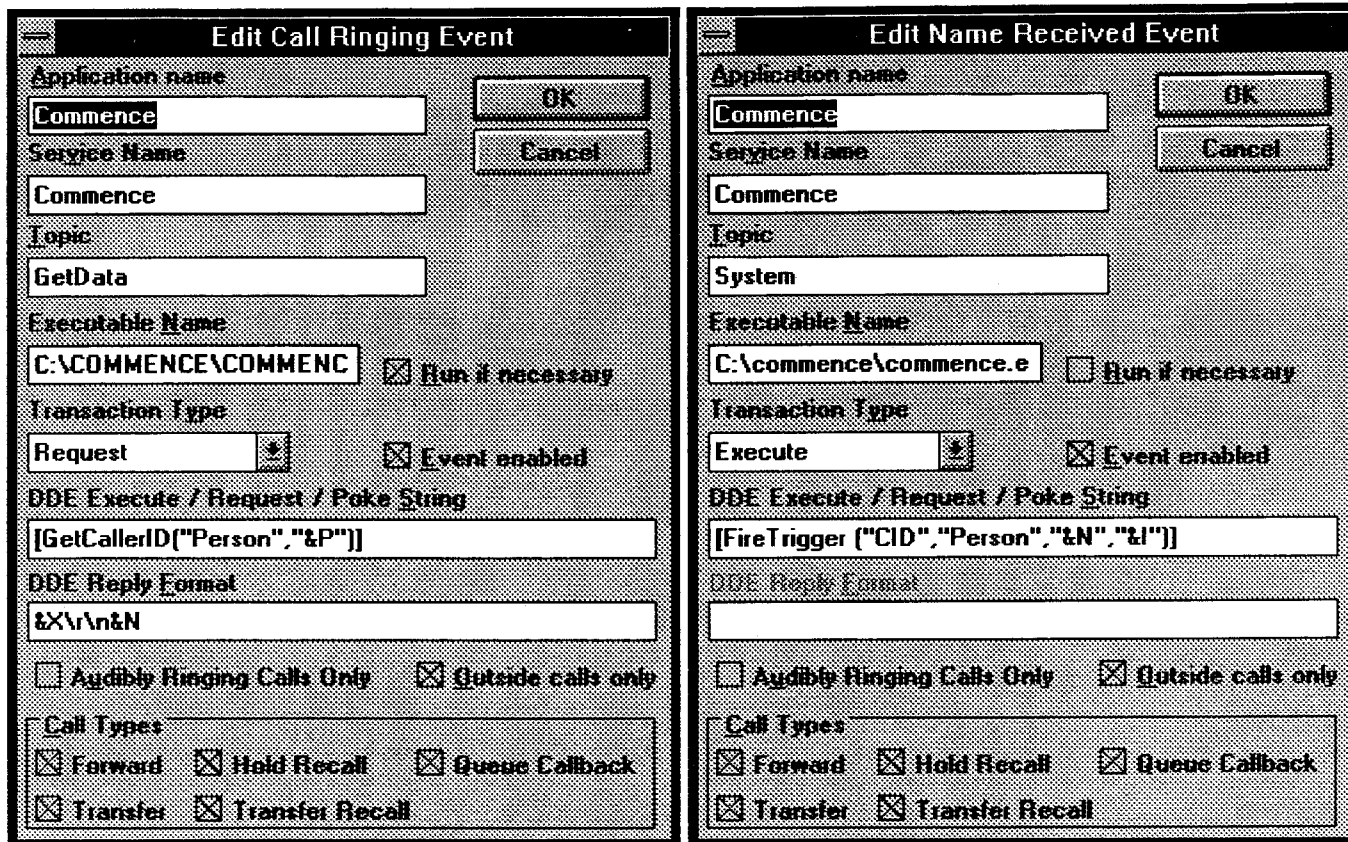
Note: If the *Commence* events have already been installed, you will get an error message and selecting OK will overwrite the existing DDE events. This will remove any custom DDE events with *Commence* as an Application name. If you have created custom DDE events, do not use the same Application names listed in this window (I.E. Use Commence-Custom, etc.).



4. To see what events have been programmed for an event type select the desired event type (Call Ringing, Name Received Events) and select Edit Events. Then select the desired event from the Event list, and select Edit Event. The DDE events are executed in the order they appear in this list. This is important in cases when you have a chain of events going on between applications to get the desired transaction sequence.



5. Select Edit Event to show the format of the DDE link. See Chapter 5 (DDE Capabilities) for a complete description of the data fields. The following windows show the default *Commence* DDE Events.

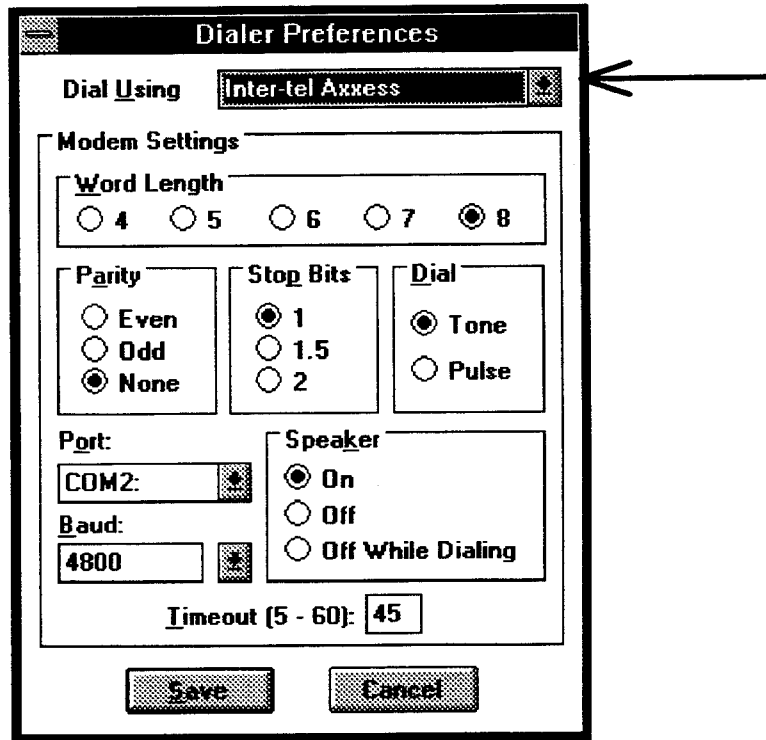


6. In the **COMMENCE.INI** file (Located in the Windows directory) add/change the following lines:

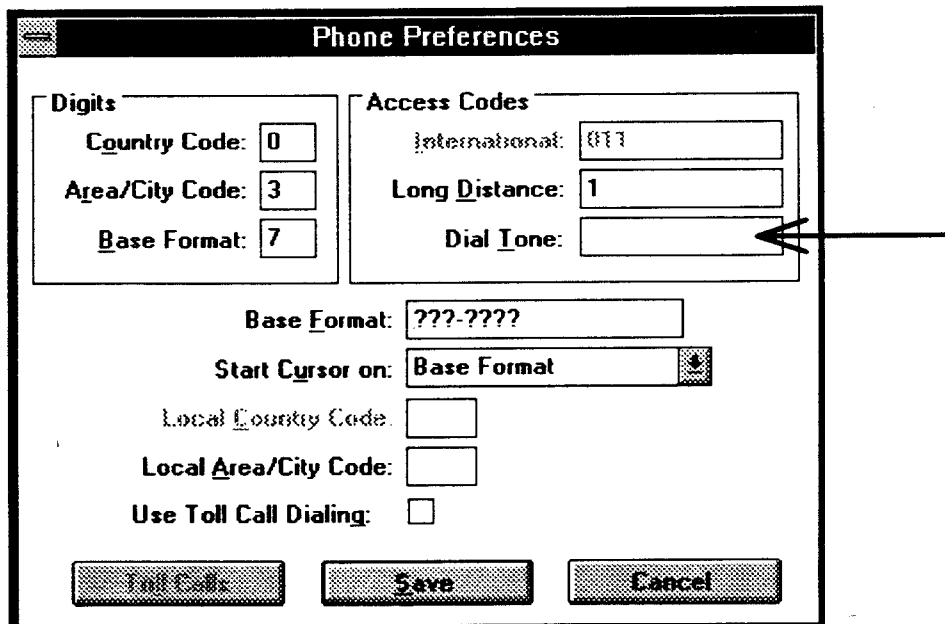
```
[Dialer]
Selected=Inter-tel Axxess
Inter-tel Axxess=[c:\axx_conn\axx_conn.exe][AXXCONNECT][System][DialNumber("", "&P", "&I")]
```

NOTE: c:\axx_conn\axx_conn.exe is the location of the *AXXESSORY* Connect application.

7. In the **COMMENCE** program, under **T**ools, **P**references, **D**ialer - choose **Inter-Tel Axxess**. That is the only parameter you need to change in this window. The rest of the options are for modem dialing.

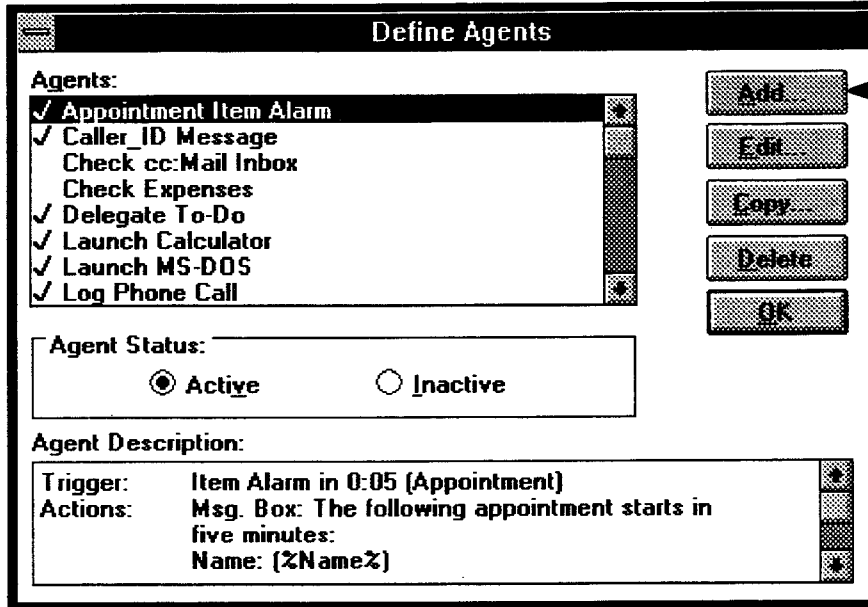


8. Remove Dial Tone parameter under Tools/Preference/Phone (unless AXXESS is behind another PBX).

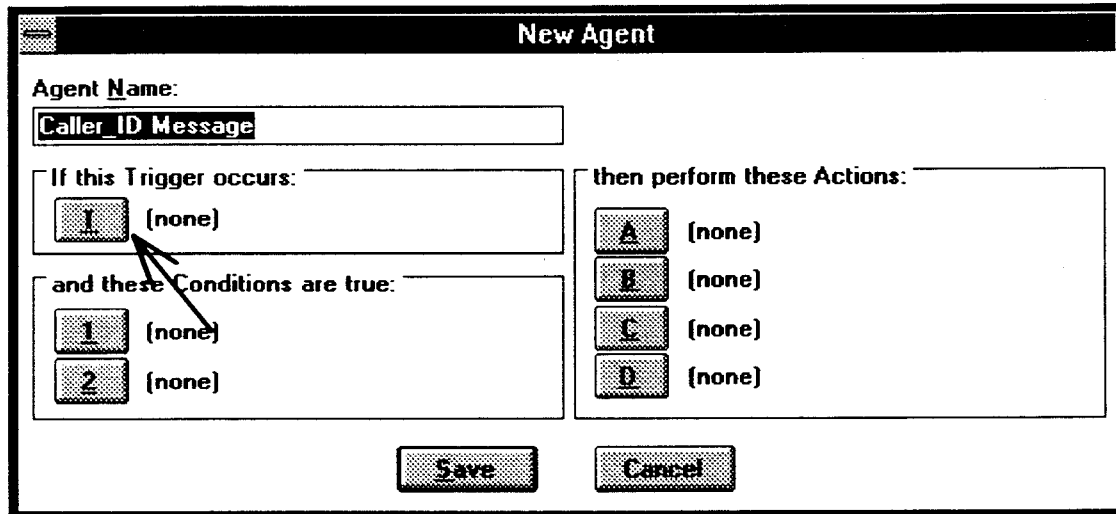


Once steps 7 & 8 are complete, you should be able to dial from *Commence* via *AXXESSORY* Connect. See the *Commence* Users guide for instructions on the various ways you can dial from *Commence*.

9. In *Commence* under Agents, Define Agents (1) Add a "Caller_ID Message" agent and (2) set up DDE links to the *AXXESSORY* Connect program:



10. This sets up a new agent "Caller_ID Message", and then select "T" to program the event when triggered.



11. Select the Receive DDE Trigger Type, set it to look for CID from *AXXESSORY* Connect, and have it match on Person field in *Commence* then select OK:

Add Trigger

Trigger Type:

- Command Bar Pick
- Item Alarm
- Menu Pick
- Receive DDE**
- Save Item
- Timer

Trigger If...

Agent Message Text Received by DDE:

CID

Enter text that would fire this trigger via the DDE Command:

[FireTrigger("CID", arg2, arg3, ...)]

arg2 equals: **Person**

and arg3 matches an item in that category.

Filtered: No **Select Filter...**

Fires upon receipt of a specified DDE message.

Clear **OK** **Cancel**

12. Select 'A' to have it edit the record it matches on (I.E. Screen-Pop):

New Agent

Agent Name:

Caller_ID Message

If this Trigger occurs:

1 Receive DDE Msg: CID

and these Conditions are true:

1 (none)

2 (none)

then perform these Actions:

A (none)

B (none)

C (none)

D (none)

Save **Cancel**

13. This sets up the record to edit, and then select OK:

Add Action

Action Type:

- Add Item
- Application Launch
- Backup
- Delete Items
- Edit Item**
- Export
- Icon Modification
- Import
- Message Box
- New View
- Open View
- Print Items
- Print Letter
- Send DDE
- Set Agent Status
- Status Bar Message
- Sync Link In

Edits one or more existing items.

Using Values From...

Trigger Condition 1 Condition 2

Selected Category: Person

Edit Item(s)...

Display Item(s) **Field Values**

Form: Initial Form

Clear **OK** Cancel

14. Select 'B' to setup a message box to give the user some options as to what to do:

New Agent

Agent Name: Caller_ID Message

If this Trigger occurs:

- 1** Receive DDE Msg: CID

and these Conditions are true:

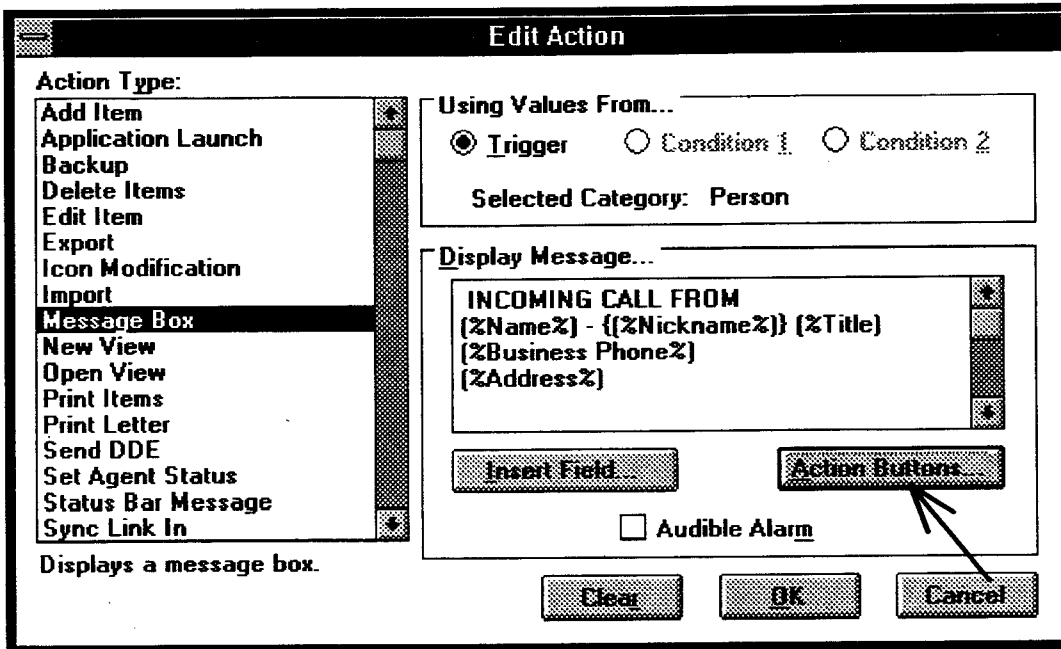
- 1** (none)
- 2** (none)

then perform these Actions:

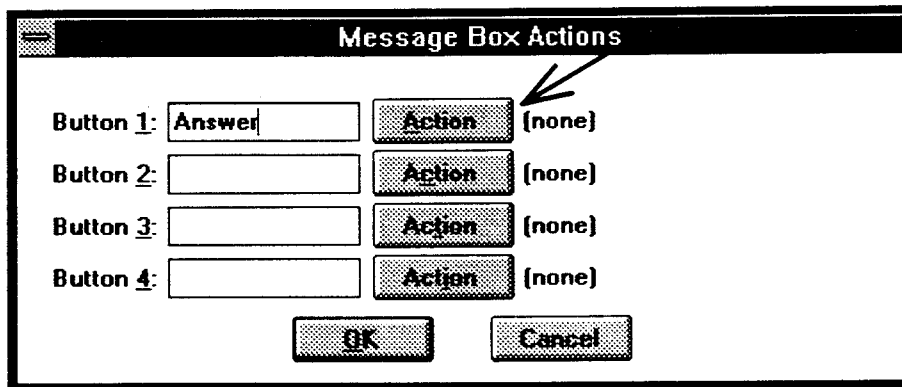
- A** Edit Item(s) in Person
- B** (none)
- C** (none)
- D** (none)

Save Cancel

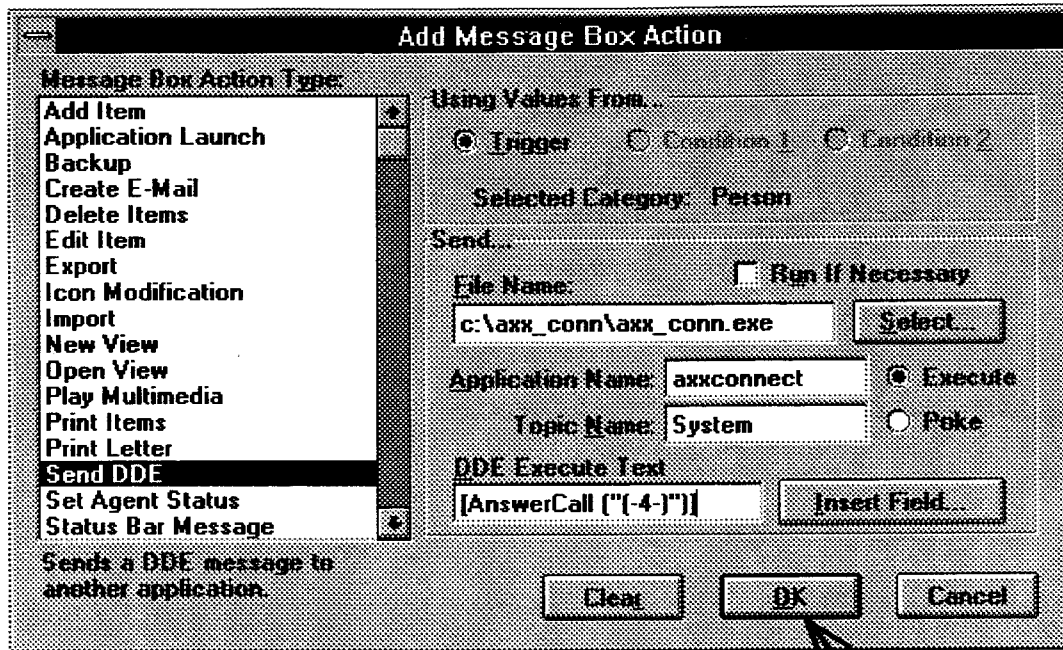
15. This defines the message box parameters. Set up the desired **Display Message** and select **Action Buttons** to define the options the user has when the call matches a record in the database. :



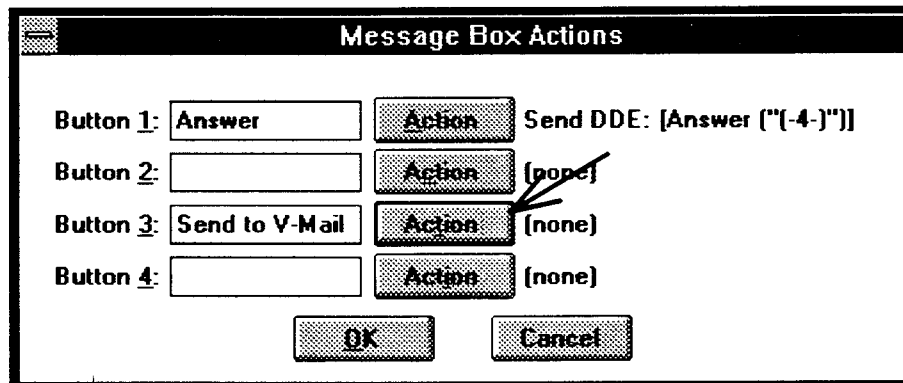
16. Add an "Answer" button as Button #1 and select **Action** to program the desired option (A DDE event to **AXXESSORY Connect** to answer the call):



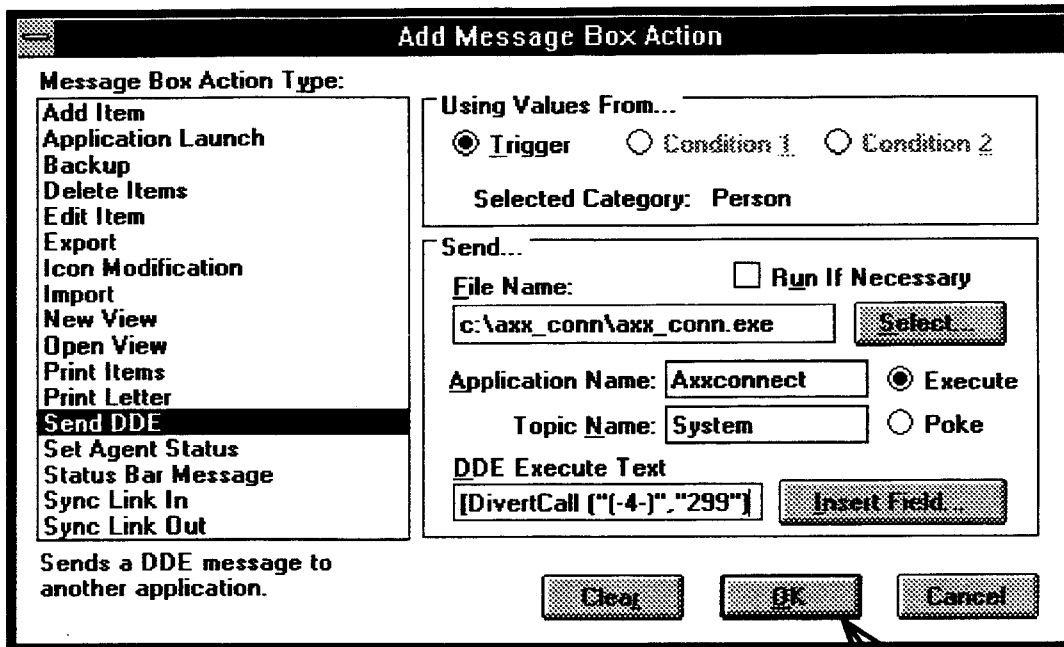
17. This defines the DDE event *AXXESSORY Connect* needs to answer the call:
 NOTE: The full path for the DDDE Execute Text is [AnswerCall ("(-4)")].



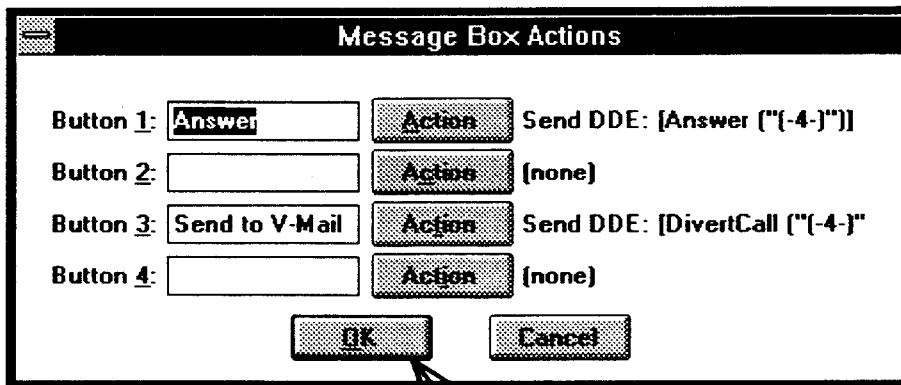
18. Add a "Send to V-Mail" button as Button #3 and select **Action** to program the desired option (A DDE event to *AXXESSORY Connect* to route the call to Voice-Mail. Substitute your voice mail access number for "299").
 NOTE: The full path for the DDDE Execute Text is [DivertCall ("(-4)","299")].



19. This defines 'DivertCall' DDE event to send the call to voice mail:



20. Select OK to save the event parameters. You will have to say OK or Save 4 or 5 times to save the event (I.E. Close all of the windows correctly). Go back in after you have saved the event to verify it was configured correctly:



Overview:

Many customers desire to use **Caller-ID** information available on incoming calls to selectively route their telephone call to specific destinations. For example, they may want calls from certain customers to go to specific sales agents. Additionally, when you subscribe to receive **Caller-ID** service from the phone company, you pay an extra charge per month per line to receive the caller's name along with the caller's phone number. A customer could potentially save a significant amount of money each month by subscribing to receive only the caller's telephone number from **Caller-ID**, and then, using his own customer database and an CTI application to lookup and provide the caller's name.

This application note describes how an attached computer could be linked to the *Axxess* phone system (Version 2.0) with a 'Desktop OAI' link, and how it could be used to lookup and provide the caller's name from the customer database and to selectively route calls to a specific destinations based on who is calling. **Note:** The *Axxess* phone system has the inherent ability to selectively route calls based on the **Caller-ID** phone number, but it may be desirable to use an attached computer in cases where (1) a large or unlimited number of customer phone numbers must be maintained, and/or (2) the customer desires to maintain this customer information themselves and have it as part of their existing 'customer database'.

Setup:

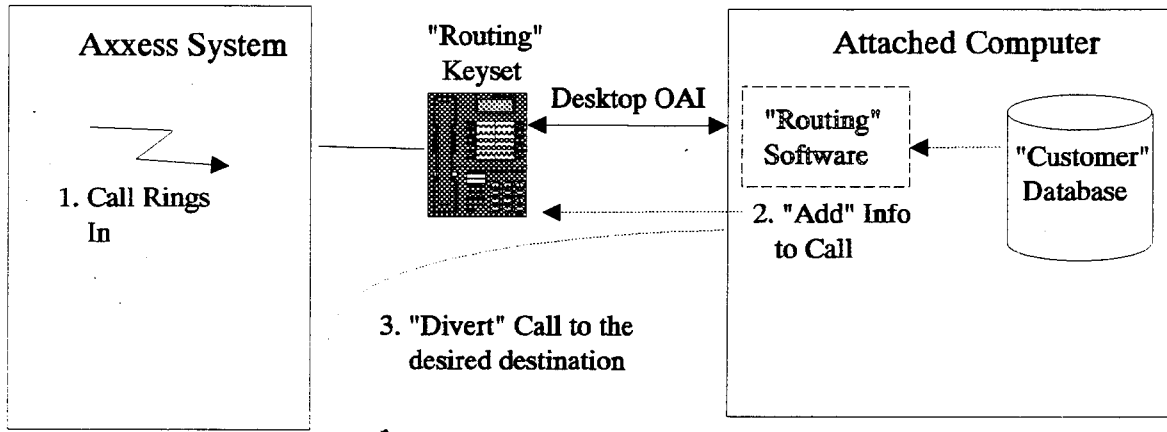
A 'Routing Keyset' equipped with Desktop OAI is connected to the Attached Computer via RS232 - this 'Desktop OAI port' is used to add the "Customer name" to the call and/or to route the call to an appropriate destination based on the calling party. Incoming calls to the *Axxess* system are programmed to ring directly and only at this 'Routing Keyset', but as a failsafe this keyset has 'Forward No-Answer' set to forward unanswered calls to a 'live operator' or 'auto attendant'. The 'Routing Keyset' would typically have its ringer 'turned-off' and would be located right next to the Attached Computer. Please refer to the attached drawing for details.

Example Scenario:

A customer, Bob Jones, calls in to "ABC Sales" and transparently his phone number is received via **Caller-ID**. The attached computer receives his phone number and recognizes him in the Customer Database as a current and frequent customer: "Name: Bob Jones", so the Attached Computer then 'adds' the name information to the call. From the customer database the Attached Computer knows that "Bob Jones" prefers to order from service agent "Martha Jones" (no relation) at extension 127, so the 'Desktop OAI' port is used to immediately transfer ("Divert") the call to her. As the call reaches "Martha Jones" desk, her computer screen can then immediately "pop" up the customer file on "Bob Jones" including what and when he last ordered. She then answers the call and says *"Well Hello Bob! I haven't heard from you since April 7th. Do you want to order some more of those ..."*

AXXESS (Version 2.0)

Caller-ID / Call Routing Application



AXXESS -- Application Note (*Desktop OAI*)

Voice-Prompted Call Routing using Desktop OAI link

10/26/94

Overview:

Sometimes **Caller-ID** information is missing or is not available on incoming calls or maybe the computer system customer database is set up to use 'customer IDs' instead of telephone numbers. This application note describes how a Voice Response Unit (Voice Computer) could be linked to the **AXXESS** phone system { using single-line ports and a 'Desktop OAI' link } to automatically answer these calls, prompt the callers to enter their "Customer Number" (or Phone Number), attach the "Customer Information" to the call, and then route the call to the desired destination. And, with the "Customer Information" now attached to the call, computer "screen pops" can now occur at each destination desktop that the call reaches.

Setup:

Single line ports from the *Axxess* system are attached to standard voice boards in the Voice Computer - these ports will be used to answer the calls, prompt the callers for information, and receive the information from the caller. A keyset equipped with Desktop OAI is also attached to the Voice Computer - this 'Desktop OAI' port is used to attach the "Customer Information" to the call and to route the call to an appropriate destination (usually based on the "Customer Information"). Please refer to the attached drawing for details.

Simplified Scenario:

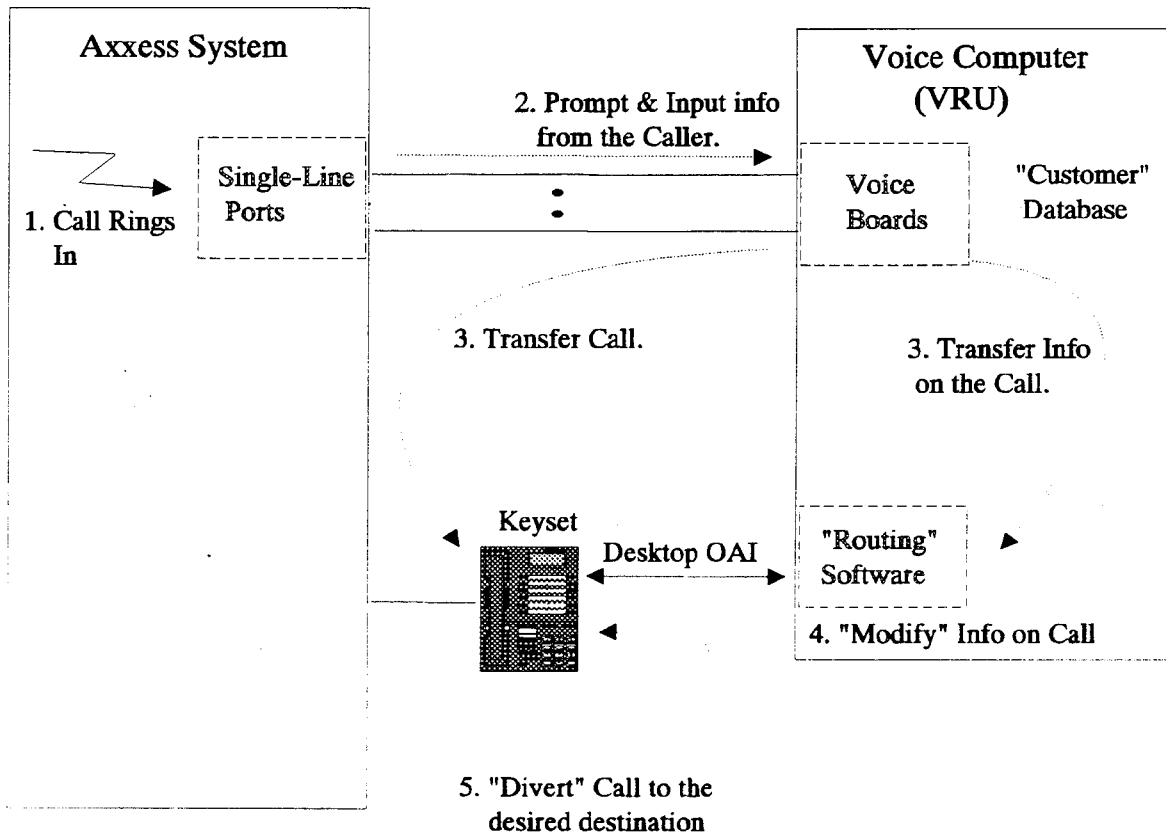
A customer calls in to "ABC Sales" and is immediately routed to a voice port on the voice computer which answers and says *"Thank you for calling ABC Sales. Please enter your 8-digit customer ID number so we can serve you efficiently."* The customer, Bob Jones, enters his customer ID and the voice computer recognizes him in the Customer Database as a current and frequent customer: "Name: Mr. Bob Jones, Phone: 602-961-9000, ID: 55512349". The Voice computer then 'moves' the call to 'Desktop OAI' port so this information can be attached to the call. Mr. Jones prefers to order from service agent "Martha Jones" (no relation) at extension 127, so the 'Desktop OAI' port is used to immediately transfer ("Divert") the call to her. As the call reaches "Martha Jones" desk, her computer screen can then immediately "pop" up the customer file on "Bob Jones" including what and when he last ordered. She then answers the call and says *"Well Hello Bob! I haven't heard from you since April 7th. Do you want to order some more of those ..."*

Advanced Scenario:

A derivative but more advanced scenario would be to have the calls ring directly into the 'Desktop OAI' port first. The voice computer could check to see if the call already has the Caller-ID information that matches with a current customer in the "Customer Database" and if so, the call could immediately be routed to the desired 'agent' without sending it to the voice prompts. If no Caller-ID information is available or the Caller-ID is not recognized (maybe "Bob Jones" is calling from a phone booth), then the 'Desktop OAI' port can immediately route the call to one of the voice response ports and the above 'Simplified Scenario' is followed.

AXXESS (Version 2.0)

Voice-Prompted Call Routing



Linking *Day-Timer Organizer* with *AXXESSORY Connect*

12/26/95

Overview:

The *Day-Timer Organizer* Personal Information Management software package is specially designed to integrate directly with the *AXXESSORY Connect* software to provide an integrated solution for (1) dialing calls out directly from the *Day-Timer Organizer* database (2) providing database lookup and 'Screen-Pop' for incoming calls ringing in with CallerID, ANI, or DNIS information, and (3) logging active calls (incoming and outgoing) from *AXXESSORY Connect* into *Day-Timer Organizer*.

Version 2.0 of *Day-Timer Organizer* and *AXXESSORY Connect* offer TAPI support, as well as the same DDE support used in version 1.0. The only change in the DDE support is in the Executable Name for Version 2 of *Day-Timer Organizer* (Default is C:\dto2\dto2.exe). In most cases, it not desirable to enable both DDE and TAPI support between *Day-Timer Organizer* and *AXXESSORY Connect*. One of the main benefits of the TAPI support option is that *AXXESSORY Connect* does not have to be active, which saves on system resources. TAPI is handled directly by the operating system. See the Application Note "Setting up the TAPI Service Provider", or the *AXXESSORY Connect V2.0 Users Guide* for TAPI setup details. This Application Note assumes that TAPI has been installed and configured. If DDE support is to be used, make sure that the Inter-Tel Desktop Telephony SP is removed, open the Control Panel, select Telephony, then remove it if it is installed. This is only for version 2.0 or greater of *AXXESSORY Connect*. You may not have a Telephony Icon in the Control Panel if TAPI was not installed.

Theory of Operation:

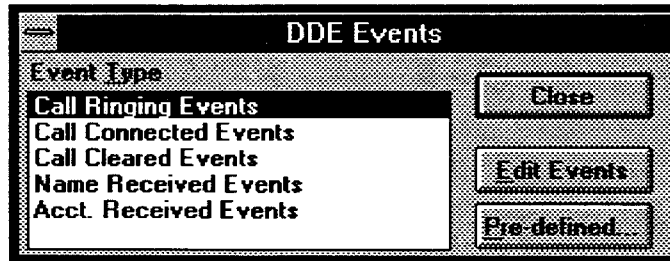
Day-Timer Organizer is setup by the user to contain an address book "database" of "contacts" (people/companies that they deal with). This database contains the names, addresses, phone numbers and other pertinent information of each contact. When placing an outgoing call to a contact, the user can simply lookup the contact in the *Day-Timer Organizer* address book and press the 'dial' key to place the call through the phone system, and provide the contact "name" to the phone system for use on display phones and system logging.

Conversely, on incoming calls, *AXXESSORY Connect* (DDE or TAPI) provides *Day-Timer Organizer* with phone call information from the phone system. *Day-Timer Organizer* then uses the 'Caller ID' information to query its address book database on the phone number fields ('Office', 'Home', etc.) and displays a list of possible callers. The user is then provided with choices of: (1) **Answer** (Answers the call in a hands-free mode), (2) **Answer, Log** (Answers the call in a hands-free mode, provides the contact's name to the phone system, and opens a 'Call Log' window to take notes on the call), (3) **To Voice Mail** (Avoids the caller by diverting the call to Voice Mail), (4) **Cancel** (DDE Only) (Closes the Day-Timer window but does not change the status of the call), and (5) **Help** (Opens the help menu to provide additional information). Note: If the user answers the call using the keyset keys or handset, they should NOT also use the **Answer** or **Answer, Log** from Day-Timer because this will cause *AXXESSORY Connect* to see another 'Offhook' request which will terminate the call. See the Steps to Setup (TAPI) section of this Application note for details on the additional call control functions available via TAPI.

Steps to Setup (DDE):

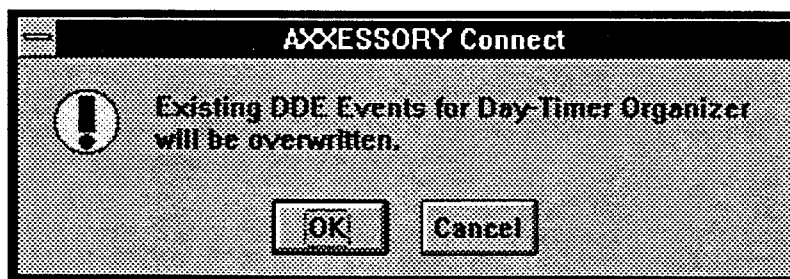
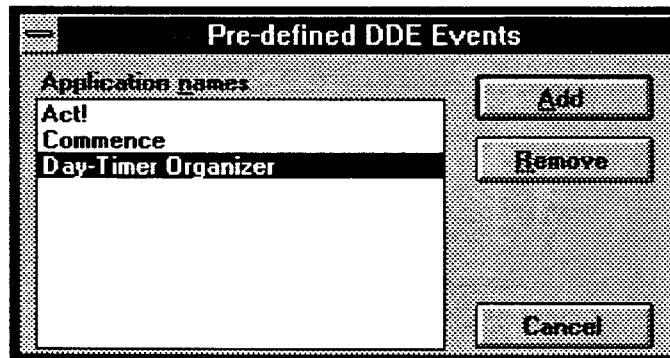
AXXESSORY Connect has standard DDE links to support *Day-Timer Organizer* for the “Call Ringing”, “Call Connected”, and “Call Cleared” Events (Setup, DDDE Events) (Application Name= *Day-Timer Organizer*, Service Name=DTO_AXXCONNECT, Topic=SYSTEM, Transaction Type=Execute). The following is a list of the steps required to install the standard DDE links for *Day-Timer Organizer*, and how to add your own custom DDE Events:

1. Under Setup on the main *AXXESSORY Connect* screen select DDE Events:
2. Highlight the desired Event Type and select Edit Events to insert/edit DDE links (See step 4), or select Pre-defined to install the standard *Day-Timer Organizer* DDE events.

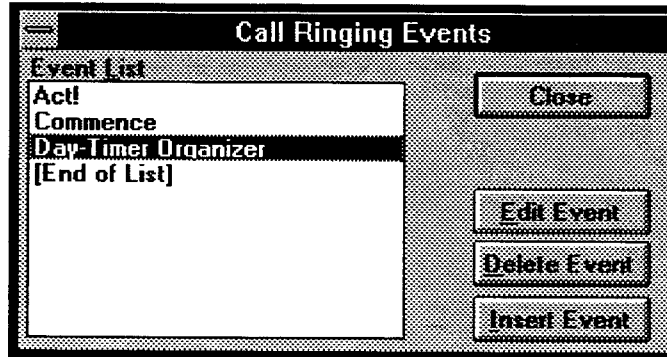


3. For Pre-defined DDE Events, select *Day-Timer Organizer* in Application names, and select Add.

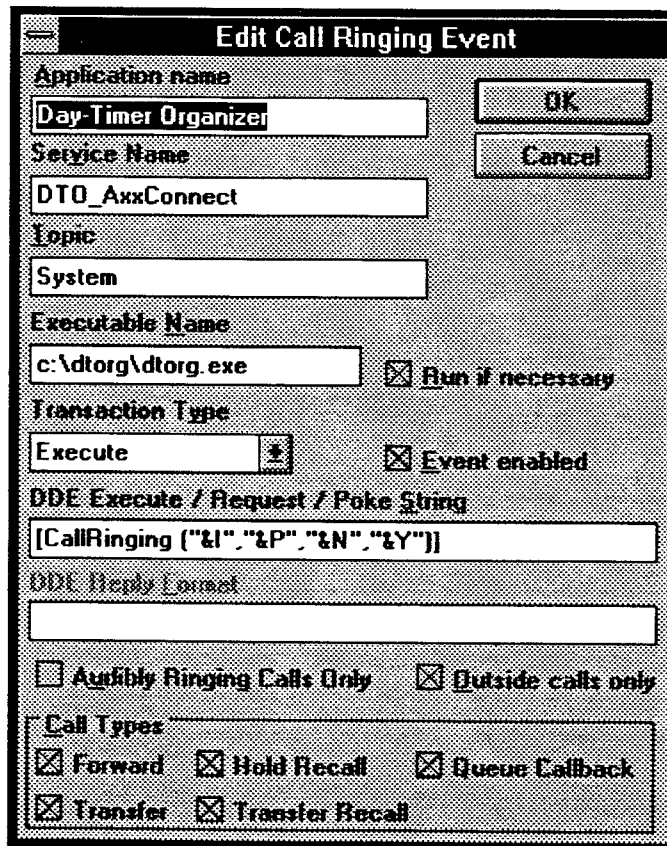
Note: If the *Day-Timer Organizer* events have already been installed, you will get the error message below and selecting OK will overwrite the existing DDE events. This will remove any custom DDE events with *Day-Timer Organizer* as an Application name. If you have created custom DDE events, do not use the same Application names listed in this window (I.E. Use Day-Timer Organizer-Custom, etc.).



4. To see what events have been programmed for an event type select the desired event type (“Call Ringing”, “Call Connected”, or “Call Cleared” Events) and select Edit Events. Then select the desired event from the Event list (**Day-Timer Organizer**), and select Edit Event. The DDE events are executed in the order they appear in this list. This is important in cases when you have a chain of events going on between applications to get the desired transaction sequence.



5. The Edit Event window shows the format of the DDE link. Refer to the *AXXESSORY Connect* manual, (section on DDE Capabilities) for a complete description of the data fields. The following three windows show the format for the data fields:



Edit Call Connected Event

Application name:

Service Name:

Topic:

Executable Name: Run if necessary

Transaction Type: Event enabled

DDE Execute / Request / Poke String:

DDE Reply Format:

Edit Call Cleared Event

Application name:

Service Name:

Topic:

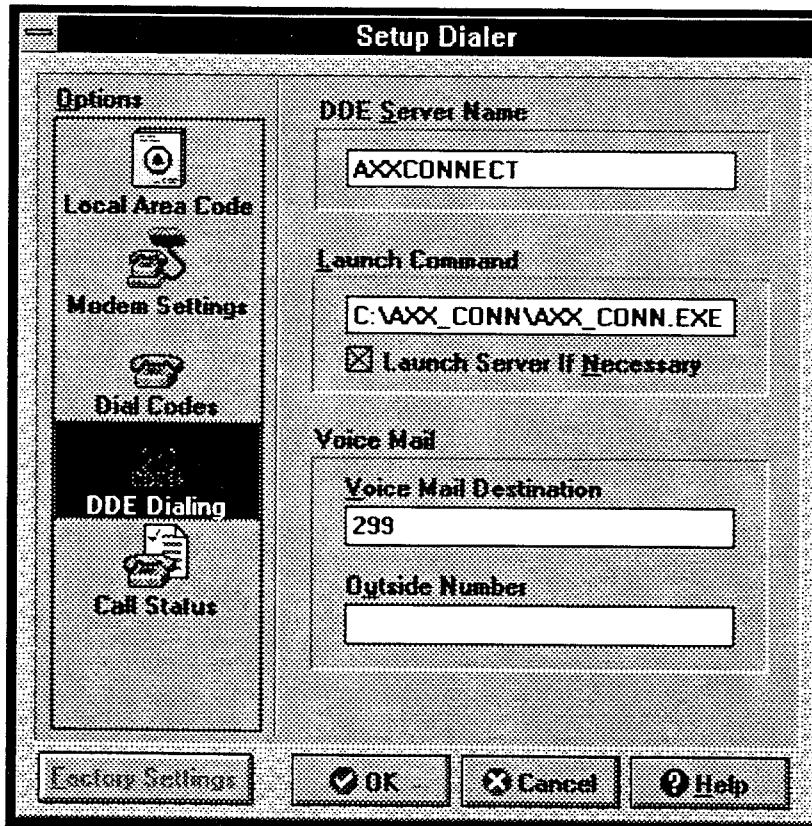
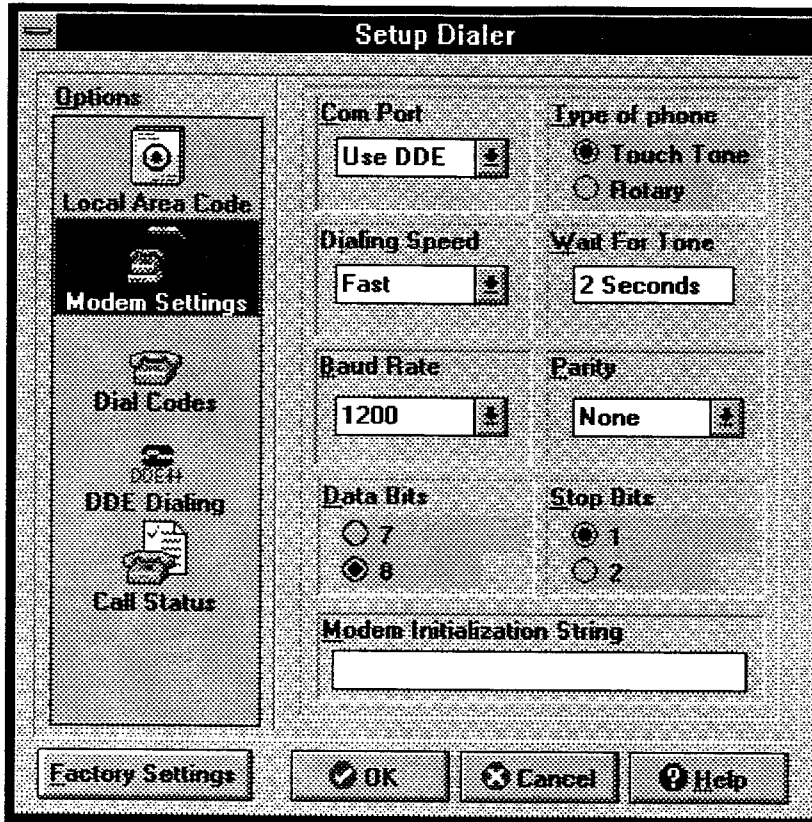
Executable Name: Run if necessary

Transaction Type: Event enabled

DDE Execute / Request / Poke String:

DDE Reply Format:

6. In *Day-Timer Organizer* select **Setup, Telephone** to bring up the Setup Dialer window shown. Select **Options - Modem Settings** and change the **Com Port** to "Use DDE" instead of a physical COM port and in **Options - DDE Dialing** program the "DDE Server Name" and "Voice Mail Destination" fields as appropriate. The Baud rate field is not really used when sending DDE messages (I.E. You do not have to match what the value is for the PCDPM in the Keyset):



7. The user will usually desire to have the ability to “log” a call (take notes on the call and save it in the contact database) even if he/she places or answers the call manually using the keyset. The procedure to add this ability to log calls from *AXXESSORY Connect* into *Day-Timer Organizer* requires adding a custom button to *AXXESSORY Connect* to send a “Log Call” DDE event. To program a custom button in *AXXESSORY Connect*, simply select **Setup, Custom Button Programming** and pick one of the unused buttons to program. Select a **Button Type** of “DDE Send”, put in a **Label** for the button, and then select **Program Event** and enter the DDE event information as shown below. See the *AXXESSORY Connect User Guide* (DDE - Capabilities, Custom Button Programming) for more information.

Program Custom Button

Button Type: DDE Send

Label: Log Call

DDE Event: [LogCall ("&I", "&P", "&N", "&Y", "&T"]

Program Event

OK, Cancel, Change Color, Log Call Sample

Edit DDE Key Event

Application name: DTO_AxxConnect

Service Name: DTO_AxxConnect

Topic: System

Executable Name: c:\dtorg\dtorg.exe Run if necessary

Transaction Type: Execute

DDE Execute / Request / Poke String: [LogCall ("&I", "&P", "&N", "&Y", "&T"]

DDE Reply Format:

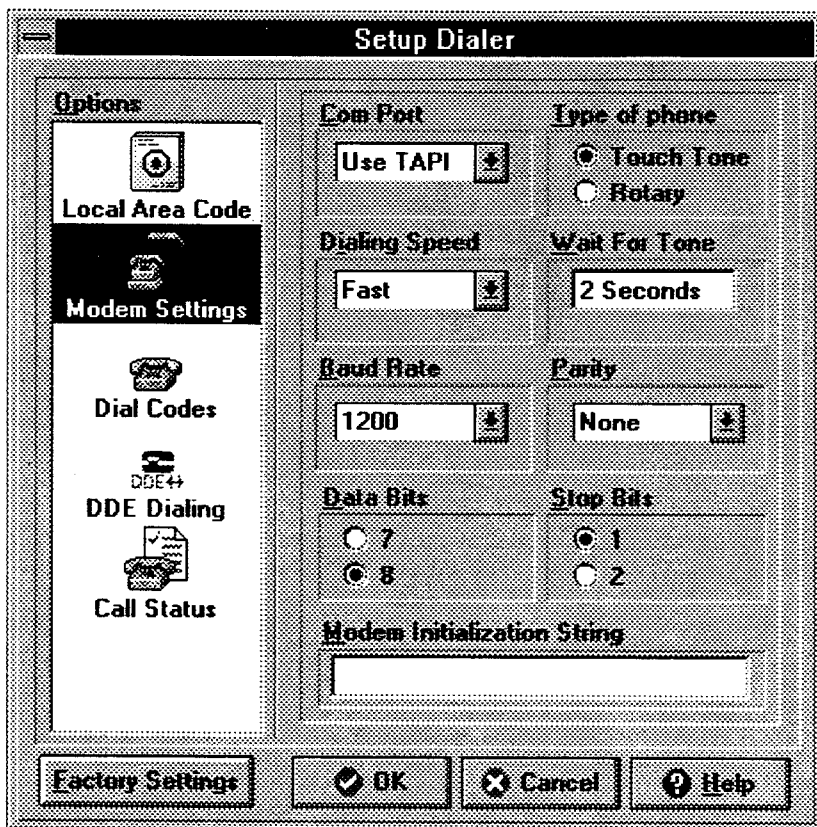
OK, Cancel

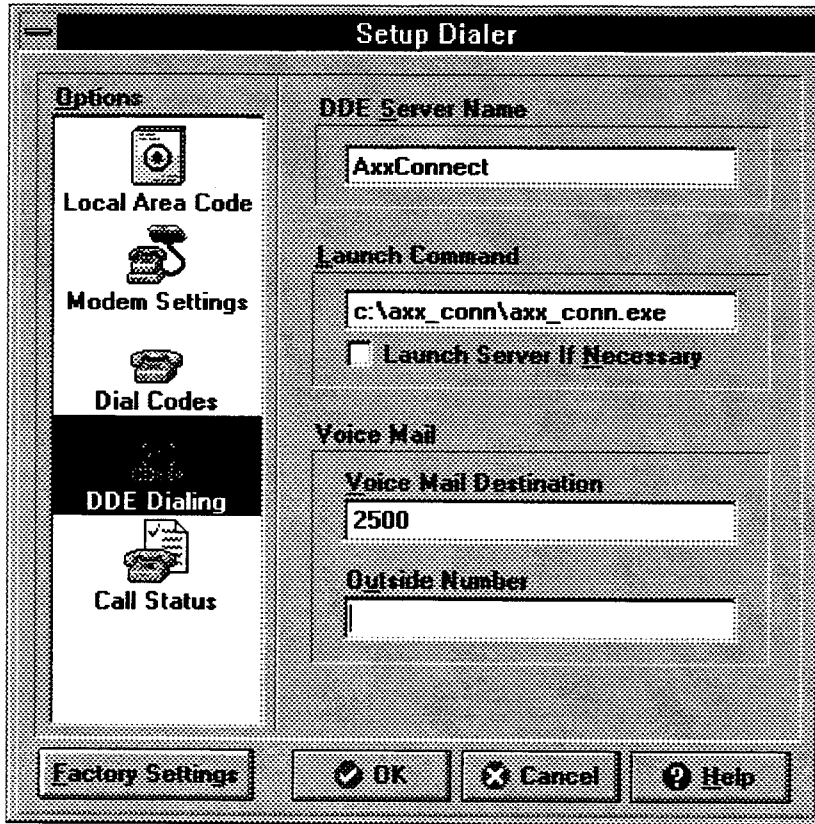
Steps to Setup (TAPI):

See the Application Note "Setting up the TAPI Service Provider", or the *AXXESSORY Connect V2.0 Users Guide* for TAPI setup details.

After Inter-Tel TAPI Service Provider has been installed and configured, the following needs to be setup in *Day-Timer Organizer* to allow TAPI outbound dialing.

In *Day-Timer Organizer* select **Setup, Telephone** to bring up the Setup Dialer window shown. Select **Options - Modem Settings** and change the **Com Port** to "Use TAPI" instead of a physical COM port and in **Options - DDE Dialing** program the "Voice Mail Destination" fields as appropriate I.E. put the extension number for Voice Mail (default is 2500). The Baud rate field is not really used when sending TAPI messages (I.E. You do not have to match what the value is for the PCDPM in the Keyset):

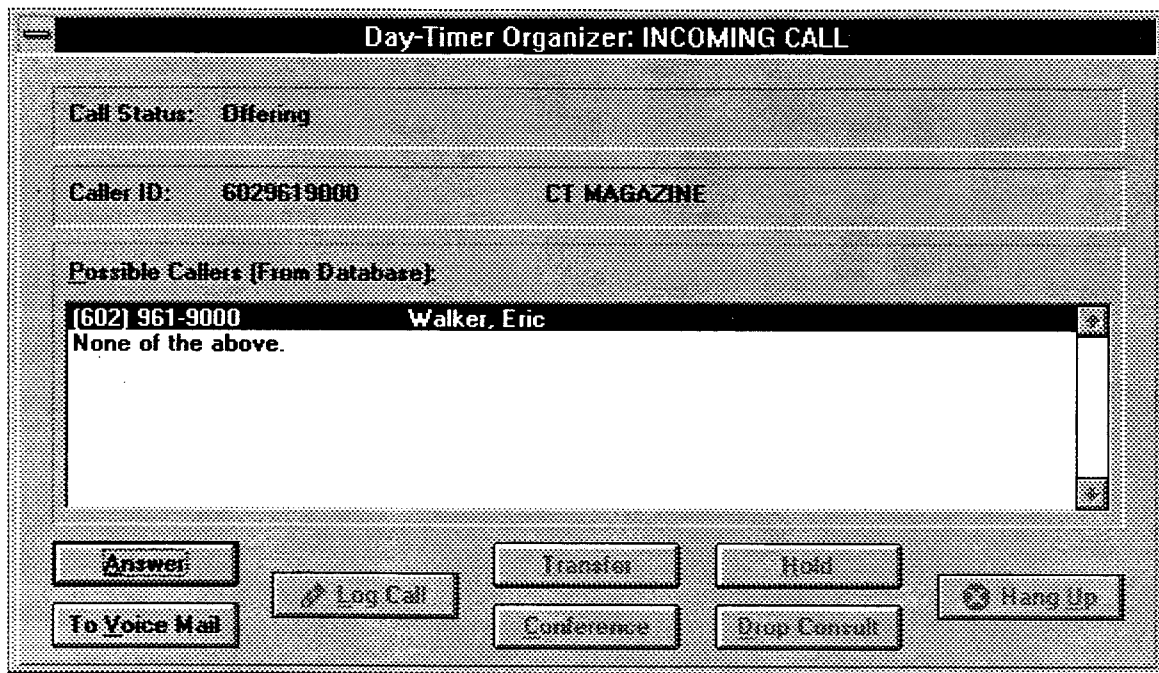




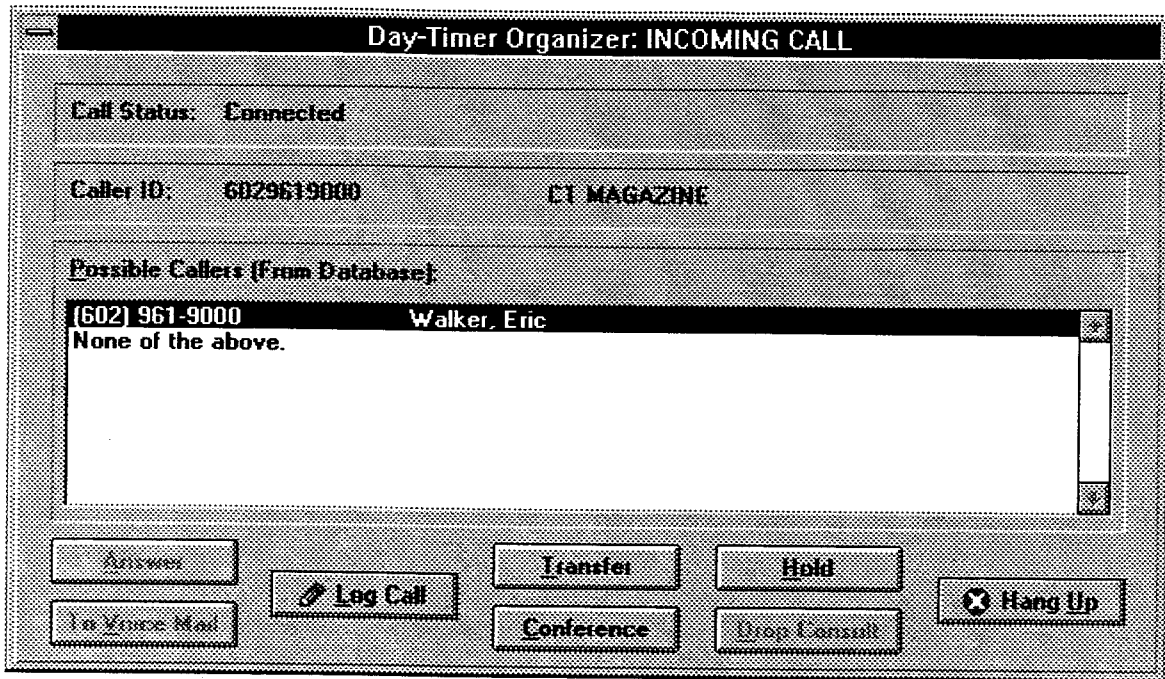
TAPI Operation:

The following screens show the TAPI call control options:

As a call rings-in, a TAPI event is sent by the **Inter-Tel TAPI Service Provider** which causes *Day-Timer Organizer* to do a Screen-pop displaying the incoming Caller-Id number and name if sent. The program provides a list of possible callers based on matching the Caller-Id. Note: In this example the Caller-Id name does not match. The user can Answer the call or Divert it to Voice Mail.



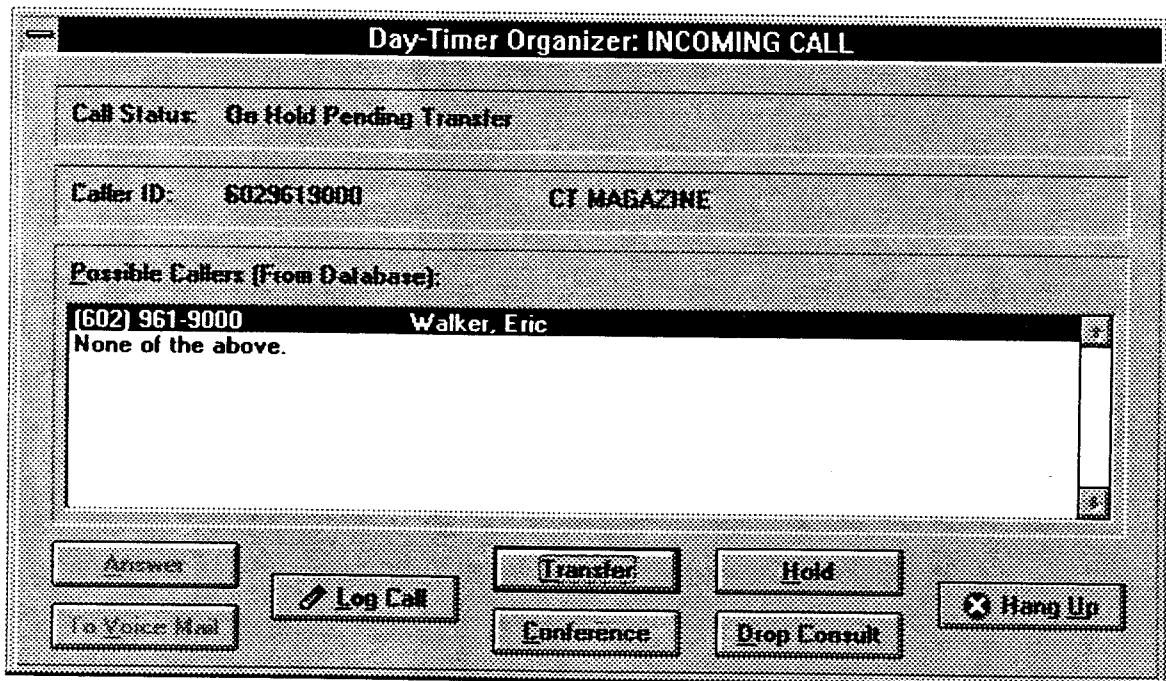
Once the call has been answered, the following call control options are available: Hang Up, Hold, Transfer, Conference.



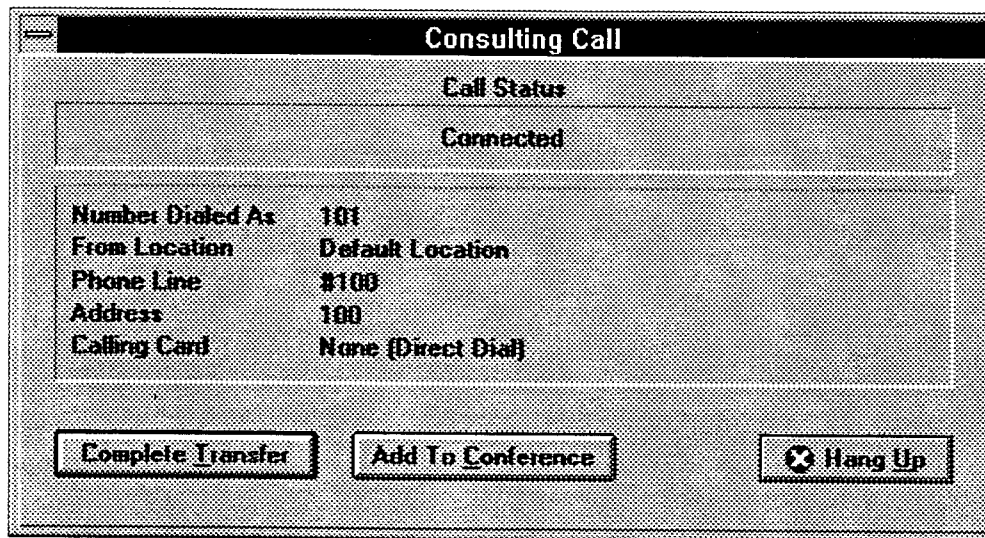
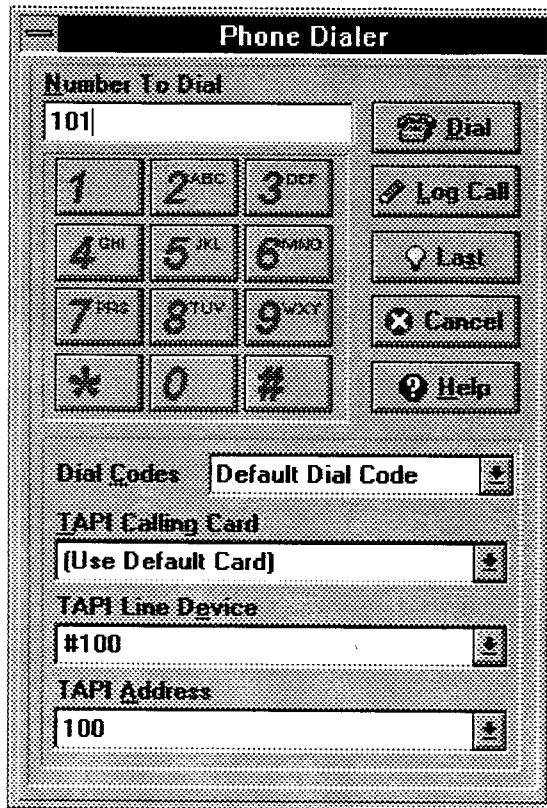
Hang Up will end the call. Hold will place the call on hold then the button will change to Un-Hold to retrieve the call.

Here are the steps needed to transfer a call:

1. Select Transfer.



2. Enter the transfer phone number. From *Day-Timer Organizer* if you select the small green phone icon on the upper right hand side, it will present a Dial pad to input the number, then select Dial.



3. Select Complete Transfer to finish.

Here are the steps needed to Conference a call:

1. Select Conference.

Day-Timer Organizer: INCOMING CALL

Call Status: **Conferenced**

Caller ID: **6029619000** **CT MAGAZINE**

Possible Callers (From Database):

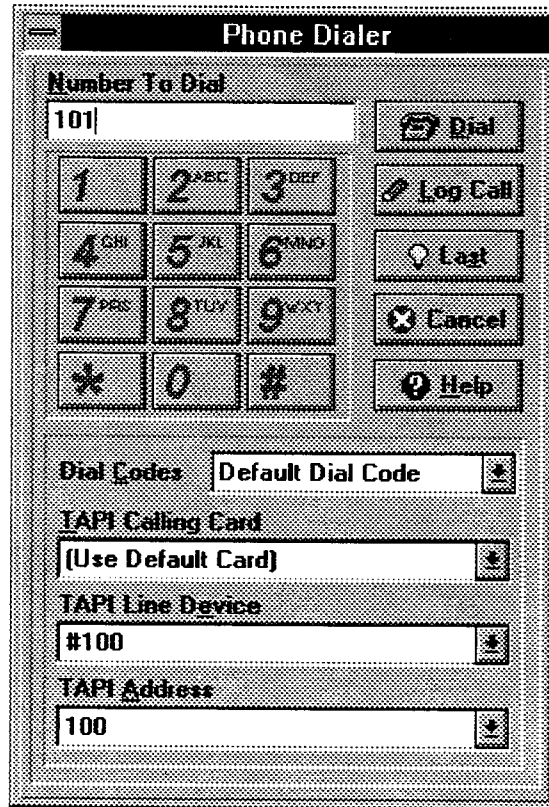
(602) 961-9000	Walker, Eric
-----------------------	---------------------

None of the above.

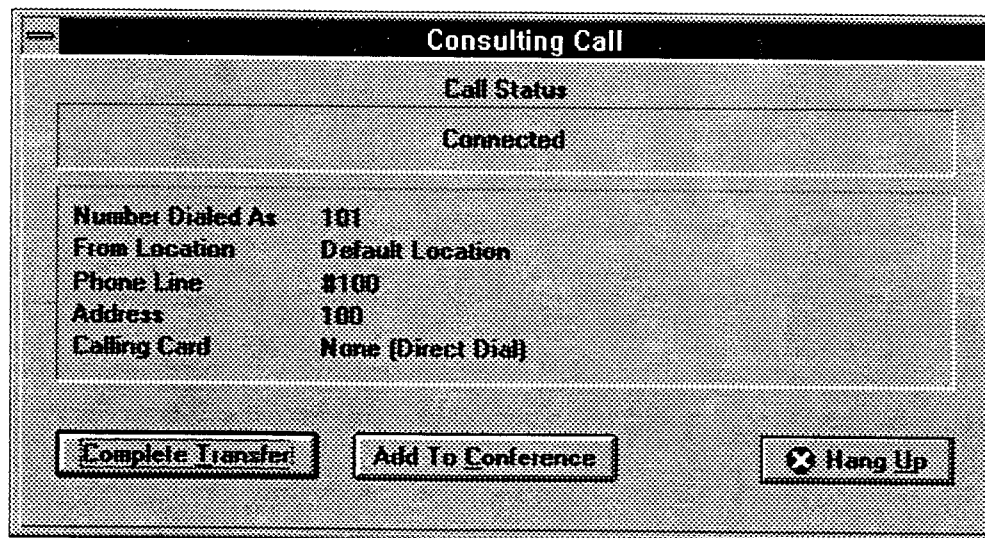
Answers Log Call Transfer Hold Hang Up

To Voice Mail Conference Drop Consult

2. Enter the next number to conference phone number. From *Day-Timer Organizer* if you select the small green phone icon on the upper right hand side, it will present a Dial pad to input the number, then select Dial.



Select Add To Conference to start the conference. Do not select Complete Transfer it will cause an error.



Select Add To Conference to start the conference. Do not select Complete Transfer it will cause an error. Note: *Day-Timer Organizer* currently only supports a three party conference. Select Hang Up to end the call.

The image shows a software window titled "Consulting Call". At the top, it says "Call Status" and "Conferenced". Below this, there is a list of call details:

Number Dialed As	101
From Location	Default Location
Phone Line	#100
Address	100
Calling Card	None (Direct Dial)

At the bottom of the window, there are three buttons: "Complete Transfer", "Add To Conference", and "Hang Up". The "Hang Up" button has a small icon of a telephone handset with a red 'X' over it.

AXXESS – Application Note (Desktop OAI)

Desktop OAI link to a PC with 'Contact Management' software

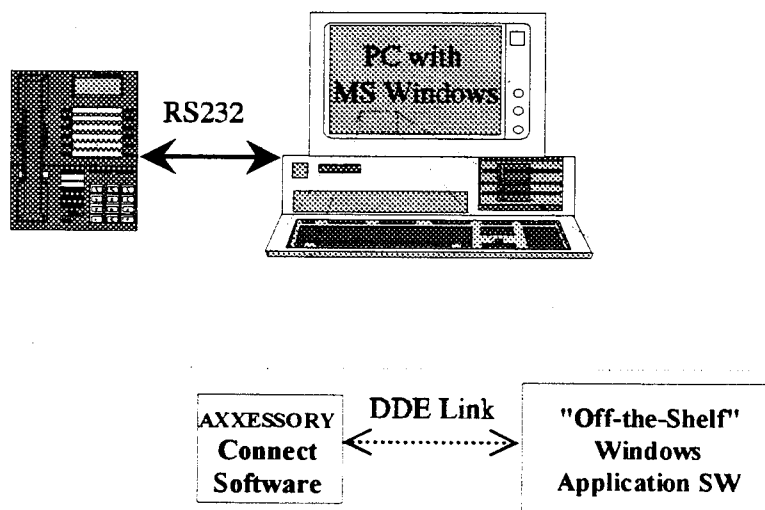
9/22/94

Overview:

This application note describes how a standard 'Contact Management' software package can be linked to the AXXESS phone system using a 'Desktop OAI' link and the productivity gains that can occur with such a setup.

Setup:

A desktop PC running Microsoft Windows is attached to an AXXESS keyset using an RS232 link from the keyset (PCDPM) to a COM port on the PC. The PC has the Inter-Tel "AXXESSORY Connect" software loaded along with an 'off-the-shelf' Windows 'Contact Management' program. The 'Contact Management' program has a complete database of customer records (with NAME, PHONE #, COMPANY, ADDRESS, RECENT PHONE CONVERSATIONS, etc.). To simplify this illustration the assumption is made that the database is contained on the attached PC, but in reality, the database could be located on another computer and accessed over a LAN or WAN from the attached PC.



Example Scenarios:

1. **Incoming Call** - A customer calls in and the call begins to ring the keyset and concurrently, through the 'Desktop OAI' link, AXXESSORY Connect is notified of the call along with the phone number of the calling customer (Caller_ID/ANI/DNIS). The AXXESSORY Connect software sends a DDE-Message to the Contact Management software saying "Calling Ringing from: 9619000" so the Contact Management software can then lookup the phone number in its database and, if recognized, POPUP a screen showing that customer's complete database record. You can then read the customer database record before answering the call and then answer more intelligently: "Hello Pete, thanks for calling back. I just checked your file and your order shipped this morning..."

2. **Outgoing Call** - The Contact Management software pops up a screen reminding you that you were to call "BOB JONES at ABC CABLE back to schedule a meeting". You decide you are ready to do this and press the "DIAL" button on the screen. The Contact Management software then looks up the phone number in the database and sends a DDE-Message to the AXXESSORY Connect software saying "Make a Call to: (602)555-1932". The AXXESSORY Connect software then pops up a "Call in Progress Screen" and sends an appropriate "Dial a Call" command over the 'Desktop OAI' link to the AXXESS phone system. The phone system automatically makes the call as if you had dialed it directly from your keyset, as you are reading though the notes from the file on Bob Jones.

AXXESS – Application Note (*Desktop OAI*)

Desktop OAI link to a PC with 'Dentist Office' software

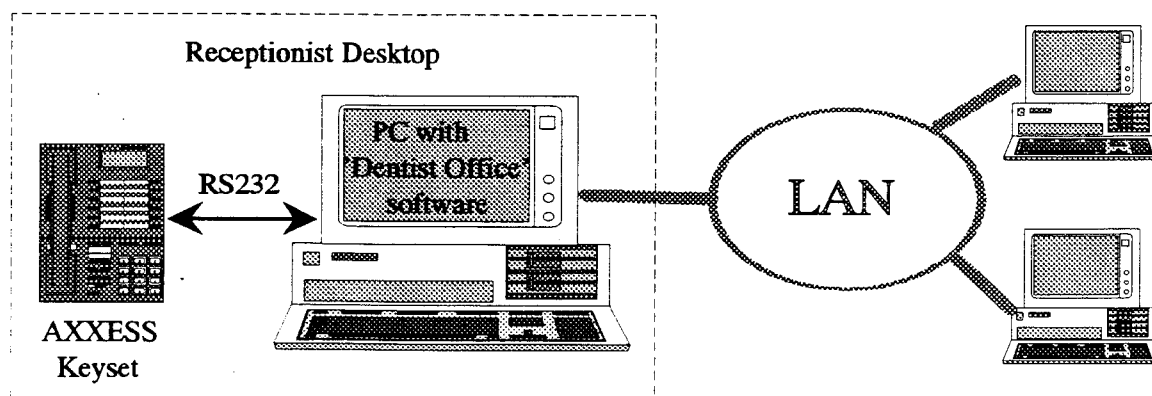
9/22/94

Overview:

This application note describes how a dentist office software package might be linked to the AXXESS phone system using a 'Desktop OAI' link and the increased productivity that can occur with such a setup.

Setup:

A PC at the receptionists desk is attached to the AXXESS keyset using a RS232 link from the keyset (PCDPM) to a COM port on the PC. The PC is running a 'Dentist Office' software program that has a complete database of all their patients records (including NAME, ADDRESS, HOME/WORK PHONE #s, DENTAL HISTORY, etc.), and an 'Appointment Scheduling' feature to keep track of all upcoming patient appointments and dentist/hygienist availability (vacations/days off/etc.).



Example Scenarios:

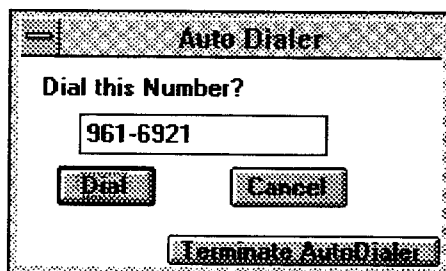
1. **Incoming Call from a Patient** - A call rings into the receptionist keyset and, through the 'Desktop OAI' link, the 'Dentist Office' software is immediately notified of the call along with calling patient's telephone number (CallerID/ANI/DNIS). The 'Dentist Office' software looks up the phone number in its patient database and pops up a screen showing the complete database record for the calling patient. The receptionist can then answer intelligently like: "Thanks for calling back Mrs. Jones. The doctor wanted us to remind you to premedicate for your heart condition two hour before coming in for your appointment tomorrow. He has already phoned in your prescription to your favorite pharmacy..."

2. **Follow-up Call** - Using the 'Dentist Office Appointment Scheduling' software, the receptionist has just canceled an appointment for Mr. Brown. Four other patients had previously expressed an interest in moving up their appointment date if any openings came available. When the cancellation occurs, the 'Appointment Scheduling' immediately finds the first of those four patients (Mr. Smith) and pops up a screen on the receptionist's PC to "Dial Mr. Smith for a Appointment Reschedule?..." and the receptionist presses the DIAL button to acknowledge and start the dialing. Then if Mr. Smith is unavailable or doesn't want the new appointment date, the 'Appointment Scheduling' would go on to the next of the four patients on the list, and so forth.

AXXESS -- Application Note (*Accessory CONNECT*)

Linking Norton Desktop for Windows 3.0: ScriptMaker with *AXXESSORY Connect* 9/22/94

Using the script language provided with Norton Desktop for Windows you can write small Windows 'applications' that are helpful in integrating other applications with *Accessory Connect*. Attached is a simple *AutoDialer* application that allows a telephone number to be **Highlighted** in another Windows application (for example in *NOTEPAD*), and then by pressing a 'Hot key' (F12), the telephone number is automatically copied to and displayed by the *AutoDialer* program.



If the **DIAL** button is then pressed, that phone number is then sent (via DDE message) to *Accessory Connect* to be dialed.

The source code for this application is attached as follows:

```

'
' *****
' * AUTODIALER Program *
' *****
'
' This program is written using NORTON DESKTOP for WINDOWS (ver 3.0)
' in the "ScriptMaker" language. It is a 'Dialer' program that can
' 'Cut/Paste' a phone number from another Windows applications and then
' send the appropriate DDE message to the "AXXESSORY Connect" to place the call
'
Sub Main()
  Dim DDEChnl As Integer
  Dim DDEcmd$
  Dim KeyPressed
  Dim TelNum$
  Dim DialMsg$
  Dim DebugMsg$

  Begin Dialog SkipsDialer 56,62,120,64, "Auto Dialer"
    PushButton 11,33,28,12, "&Dial"
    TextBox 17,17,70,12, .TextBox1
    Text 6,5,61,8, "Dial this Number?"
    CancelButton 60,33,33,12
    PushButton 42,54,77,9, "Terminate AutoDialer"
  End Dialog

  Dim DialBox as SkipsDialer

  'Put up an Initial Information Box
  DialMsg$ = "AutoDialer: From any Windows Application highlight the phone number you
  want to dial then press F12. Or Press F11 to dial a number that has been cut to the
  Clipboard."
```



```

MsgBox DialMsg$, 0, "Auto Dialer"

DialLoop:
  QueEmpty
  KeyPressed = WaitForKey("{F12}", "{F11}", "{F12}", "{F12}", "{F12}" )
  TelNum$ = Clipboard$()

  If KeyPressed = 1 Then 'If "F12" pressed...
    SendKeys "^{INSERT}^C" 'Send a "Copy" command to active application
  End If
  DialBox.TextBox1 = Clipboard$()
  KeyPressed = Dialog(DialBox)
  TelNum$ = DialBox.TextBox1

  If KeyPressed = 2 Then 'The "Terminate" key was Pressed
    Exit Sub ' so Exit the Program...
  End If
  If KeyPressed = -1 Then 'The "Enter" Key was pressed in Dial Buffer
    KeyPressed = 1 ' Change to show "DIAL" key pressed
  End If

  'If "DIAL" button pressed and there is a phone number...
  If KeyPressed = 1 And Len(TelNum$) > 0 Then
    'Build and Send a DDE message to AXCESSORY Connect
    DDEcmd = "[DialNumber("", "")"
    DDEcmd = DDEcmd + TelNum$ + "", "", "" ]"
    DDEchnl = DDEInitiate("AXXCONNECT", "SYSTEM" )
    If DDEchnl > 0 Then
      DDEExecute DDEchnl, DDEcmd
      DDETerminate DDEchnl
    Else
      MsgBox "AXCESSORY Connect isn't Running", 1, "Auto Dialer"
    End If
  End If

  End If
  'Stay in an Endless Loop
  Goto DialLoop

End Sub

```

Linking Hayes-Dialing Programs with AXXESS (*Desktop OAI*)

9/22/94

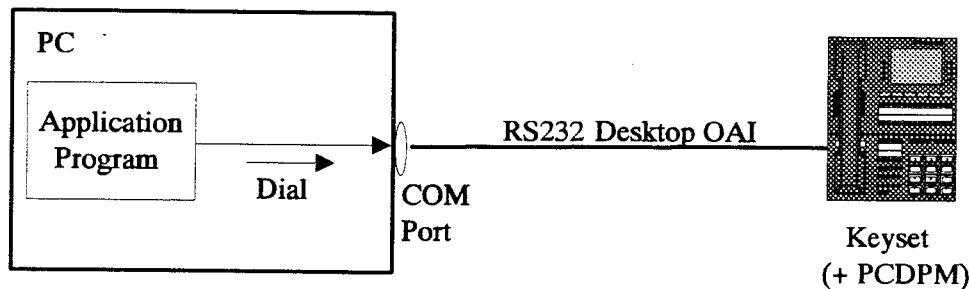
Overview:

A wide variety of application programs (i.e. PIM and Contact Management programs) exist on the market that do out-bound dialing using a Hayes-compatible Modem. These application programs exist on many computer and operating systems platforms (i.e. DOS, Windows, Unix, NextStep, etc.), and are a very efficient means for using a computer database (or in effect a 'Rolodex') to quickly and accurately place telephone calls.

Since the AXXESS desktop OAI link supports Hayes-compatible commands, these same off-the-shelf application programs can very often be directly connected to an Axxess keyset (thus eliminating the need for a modem and an additional analog phone on each desktop), thus integrating the automated dialing directly with your desktop electronic keyset. This application note describes the steps required to setup the Desktop OAI Interface in the AXXESS System and the general setup required for the application program to do out-bound dialing.

Following is a current list of programs we have tested and verified:

{ Windows Cardfile, Goldmine for DOS, ECCO, Sidekick, Lotus Organizer, Jensen-Jones StartUp, PackRat, Telemagic for DOS, Instant Recall, Ascend 4.0, ACT! }



Steps to Setup:

A. The AXXESS system:

Note: See the *AXXESS Installation and Field Maintenance Manual* for complete details on the setup procedure.

1. The AXXESS System must be running version 2.0 software or greater with the Desktop OAI Interface premium feature enabled.
2. A PCDPM must be installed in the Keyset (Phone).
3. The station circuit for the Keyset must be configured for use with a 'Desktop Interface'.
4. From the Keyset, enter the Program Baud Rate (393) Feature Code to change to the desired value. The current Baud rates supported are: 300, 600, 1200, 2400, 4800.

B. On the PC:

1. Install the desired Application software on the PC.
2. Attach the PCDPM RS232 cable to the COM port.
3. In the application software, configure the COM port and Baud rate desired. Some of the programs-tested do not have the ability to change their Baud rates, and they may be hard to determine which rates they support (usually 1200 is OK). Adjust the Baud rate on the Keypad as required.

NOTE: Most of the programs we tested send a standard Hayes "Hangup" command (ATH0) when you press the 'Call Answered' button (or equivalent) as these programs are intended to be used with a modem AND a standard analog phone in a 'Y' configuration, and when the handset of the analog phone is picked up, they want to 'Hangup' the modem to avoid loading down the phone line. But, when directly connected to the *Access* keypad using Desktop OAI, honoring this standard "Hangup" command would mistakenly 'Hangup' the keypad and thus terminate the call -- certainly an undesirable affect. Therefore, the Desktop OAI ignores the standard Hayes "Hangup" command (ATH0) but instead responds to an alternative "Hangup" command (ATH3).

SECTION 5: Development Tools

Since the *Axxess Desktop OAI Link* uses a standard RS232 port from the attached keyset (using the PCDDPM) and since the protocol is implemented using the 'visible' section of the ASCII character set, it is very easy to simply connect it up to a standard COM port of a PC. Using an off-the-shelf terminal emulator (like TERM provided with MS-Windows), you can easily type in commands and monitor events back from the keyset. To assist you in this regard, we've included a MS-Windows TERM configuration file created specifically for use on the Desktop OAI link. This is included on the "**Desktop OAI Tools**" floppy.

Also included on the "**Desktop OAI Tools**" floppy is an "*Axxessory Connect: DDE Simulator*" program that is useful when you're developing an application program (or integration) that will use the DDE interface of the *Axxessory Connect* when you don't have an Axxess system, Axxess keyset + PCDDPM, and *Axxessory Connect* software all available. This simulator runs under MS-Windows and can simulate the generation of some of DDE events that would come from the *Axxessory Connect* when calls ring in, are answered, and 'clear' out.

Another tool included on the "**Desktop OAI Tools**" floppy is an "*Axxessory Connect: DDE Editor*" program. When you have developed a program to work with *Axxessory Connect* using DDE, this tool can be used to build a special DDE integration file that makes it very simple for an *Axxessory Connect* user to set up the DDE links to your program once it is installed.

As mentioned previously, an "On-Line Help" file is also provided on the "**Desktop OAI Tools**" floppy with all of the OAI protocol information from Section 3 of this toolkit, thus providing an on-line reference tool.

AXXESSORY CONNECT: DDE Simulator

1.0 Overview:

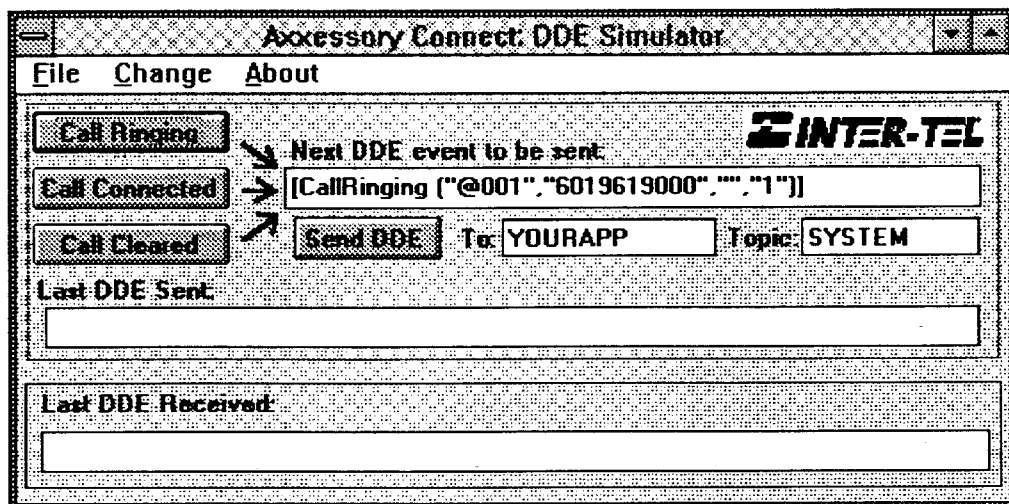
This program, called *DDESIm*, is a MS-Windows based program that can 'Simulate' some of the DDE interface of the AXXESS *Accessory Connect* (PC-Based Telephone), so that application programs can be developed and tested without having an AXXESS system, keyset, and *Accessory Connect*.

2.0 Installing *DDESIm*:

The program can be installed using MS Windows program manager by selecting **R**un and the filename: **SETUP.EXE** from the installation floppy disk. Once installed under windows you can simply double-click on the *DDESIm* Icon to run the program.

3.0 Using *DDESIm*:

The top section of the DDE simulator screen pertains to DDE events that are to be sent to the 'target' application program. Three types of DDE events (**Call Ringing**, **Call Connected**, and **Call Cleared**) are preloaded into the DDE Send buttons. Pressing one of these three buttons will put the selected DDE event into the "Next DDE event to be sent:" window, but doesn't send the DDE event. Here the text and parameters of the DDE event can be changed before it is sent. Pressing the **Send DDE** button will copy the DDE message to the "Last DDE Sent:" window and will attempt to deliver the DDE message. If the DDE message is not received by the target application, and appropriate Error message window will popup to explain why.



The bottom section of the DDE Simulator screen has a window labeled "Last DDE Received:" . Any DDE messages sent to the DDE simulator from another application program will show up in this window preceded with a Date and Time stamp to show when it arrived. (NOTE: To send a DDE message to the DDE simulator, it should be sent to Application: **DDESIM** and Topic: **SYSTEM**).

The *DDEsim* program can also be used to try out DDE messages to/from the *Axxessory Connect* program. For example, you may want to try sending a "Dial Number" DDE event to *Axxessory Connect* to experiment with how the parameters of that command work, or you may want to see the format of the "Call Ringing" DDE event from the *Axxessory Connect* program.

4.0 Changing DDE parameters:

When you change a DDE Send Button (from the **C**hange menu, **B**utton 1, **B**utton 2, **B**utton 3) you should set up the **A**pplication Name: and **T**opic: where the DDE events will be sent. Typically the Application Name is the Executable file name without the .EXE extension (i.e. **AUTOFWD** for the **AUTOFWD.EXE** program), but be aware that this is not always the case. Also the text of the **DDE String:** can be changed from this screen which will then be used whenever the respective DDE button is pressed.

The programming of the "DDE Send Buttons" simply defaults the Text, Application Name (To:), and Topic of those respective DDE event messages. When you press a DDE Send Button, these fields are loaded onto the screen, can then be changed to virtually anything, as in the following example that shows how you configure it for sending a "Dial Number" event to *Axxessory Connect*.

The screenshot shows a dialog box titled "Edit DDE Event". It contains the following fields and controls:

- Application Name:** A text box containing "AXXCONNECT".
- Topic:** A text box containing "SYSTEM".
- DDE String:** A text box containing "[DialNumber (\"\", \"6019619000\", \"Inter-Tel\"]".
- Button Label:** A text box containing "Dial".
- OK** and **Cancel** buttons.

When you **S**ave or **E**xit from the **F**ile menu, this information is saved so it is retained for the next time the DDE simulator is used.

AXXESSORY Connect: DDE Editor

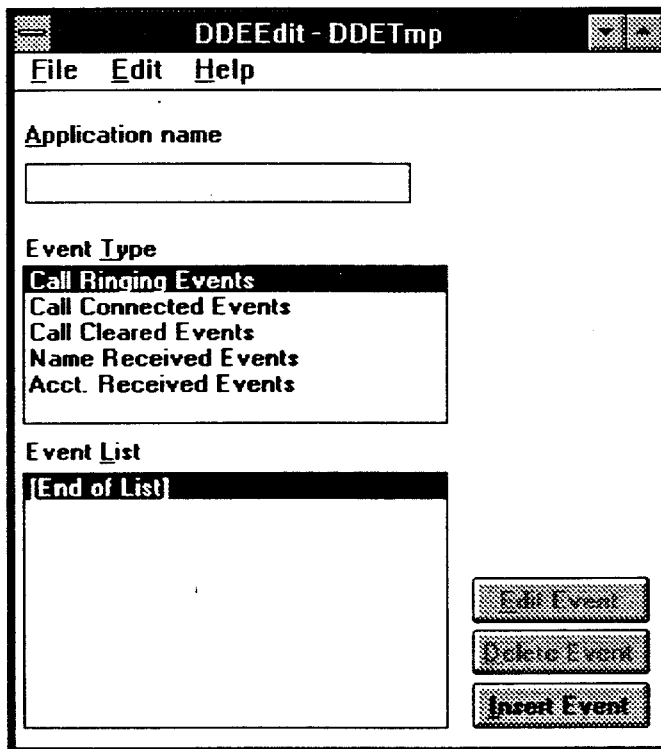
Overview:

This program, called *DDEedit*, is a MS-Windows based tool that allows third-party developers to create DDE Event “templates” for *Axxessory Connect*. These templates can then be made available to *Axxessory Connect* end users to greatly simplify the DDE configuration process for third-party applications.

Installing *DDEedit*:

The program can be installed using MS Windows program manager by selecting **R**un and the filename: **SETUP.EXE** from the installation floppy disk. Once installed under windows you can simply double-click on the *DDEedit* Icon to run the program.

Using *DDEedit*:



The DDE Editor operates on *Axxessory Connect* DDE Template files, which have an extension of **.DDT**. Files can be created, opened and saved using the familiar New, Open, Save and Save As commands from the File menu.

When it is executed, the DDE Editor will create a new, empty template as shown above. The significance of each field is explained below:

Application name

This is the application name the user will see when he / she clicks on the "Pre-defined" button in the DDE Events dialog of the *Axxessory Connect* (see the *Axxessory Connect* User's Manual for more details). It will also show up in the Application Name field of the DDE Event Edit dialog. Since it is used for information purposes only, it does not necessarily need to be the same as the DDE Service Provider name (See "Editing Events" below).

Event type

There are five different types of DDE Events that the *Axxessory Connect* is capable of generating. Selecting an event type from this list will display the events of that type in the Event List list box.

Event list

This list box shows all the DDE Events for the event type selected in the event type list box. An empty list will contain only the "[End of List]" placeholder. The text in this list box will be the same as the Execute / Request / Poke string for the given event (see "Editing Events" below). The list box is not sorted alphabetically; the events are listed in the order that they will be generated by the *Axxessory Connect*.

Edit event

Clicking on this button will edit the event selected in the event list. If "End of List" is selected, this button will be disabled. See "Editing Events" below.

Delete event

Clicking on this button will delete the event selected in the event list. If "End of List" is selected, this button will be disabled.

New event

Clicking on this button will create a new event of the type selected in the Event Type list box. See "Editing Events" below.

Editing Events:

The dialog displayed for editing events will vary slightly, depending upon which event type is being edited. The two possibilities are shown below:

Edit Event Type

Application name
[]

Service Name
[]

Topic
[]

Executable Name
[] Show if necessary

Transaction Type
Execute [v] Event enabled

DDE Execute / Request / Poke String
[]

DDE Reply Format
[]

OK
Cancel

If a "Call Connected" or "Call Cleared" event is selected for editing, this dialog will be displayed. The dialog is the same for both event types; the only difference is in the title and the default DDE Execute / Request String that will be displayed when editing a new event.

Edit Event Type

Application Name: [disabled field]

Service Name: [text field]

Topic: [text field]

Executable Name: [text field] Run if necessary

Transaction Type: **Execute** Event enabled

DDE Execute / Request / Poke String: [text field]

DDE Reply Format: [text field]

Audibly Ringing Calls Only Outside calls only

Call Types:

Forward Hold Recall Queue Callback

Transfer Transfer Recall

OK Cancel

If a “Call Ringing”, “Name Received” or “Acct Code Received” event is selected for editing, this dialog will be displayed. The dialog is the same for all three event types; the only difference is in the dialog title and the default DDE Execute / Request String that will be displayed when editing a new event.

With the exception of the Application Name field (which is disabled and will contain the same application name that was entered in the main screen), these dialogs are identical to the DDE Event Edit dialogs in the AXESSORY Connect. Refer the the *Accessory Connect* User’s Manual for more details.

Making DDE Event Templates available to *Accessory Connect*:

When the user presses the “Pre-defined” button in the DDE Events dialog while running AXESSORY Connect, the *Accessory Connect* software searches its installation directory for all files with a .DDT extension. It then checks the files to insure that they contain valid DDE Event templates and, if so, displays the Application Name (as entered using *DDEEdit*) in its list of applications.

Therefore, in order for the *Accessory Connect* software to be able to recognize a template, the template need only be copied to the *Accessory Connect*’s installation directory (C:\AXX_CONN by default).

SECTION 6: *Axxessory Connect* (DDE & TAPI)

Inter-Tel has designed an MS-Windows based product, called the *Axxessory Connect*, that connects to the **Desktop OAI** link and provides a 'PC-Based Telephone'. This may be very useful for OAI developers developing products for the MS-Windows environment, because the *Axxessory Connect* also has a standard, programmable 'DDE-interface' that provides another OAI interface in addition to the RS232 level Desktop OAI link.

That is, instead of developing an application program specifically to use the proprietary RS232 level Desktop OAI link, it may be much easier and more logical for you to develop (or simply) adapt your application to use the standard DDE links provided by the *Axxessory Connect*, and then simply 'coexist' with the *Axxessory Connect* on the desktop PC. This allows your application (and your application developers) to concentrate on its specific functions, and lets the *Axxessory Connect* provide the user interface for the telephone functions and the RS232-level functionality.

The *Axxessory Connect*, also provides a TAPI SPI (Service Provider Interface) so that TAPI-compatible applications can run directly with the Axxess and Axxent telephone systems. The detailed information on the features and capabilities provided by the TAPI SPI is included in the 2nd half of this section. For more details on TAPI and its capabilities, please contact Microsoft corporation directly.

To assist OAI developers with both of these types of development, we've included a sample working copy of the *Axxessory Connect* (on the *Axxessory Connect Disk*) and its user documentation (which includes a complete description of its DDE interface). **NOTE:** To use the *Axxessory Connect* you must have *Axxess V2.0* (or later) system software {or *AXXENT V1.0* or later}, have the **Desktop OAI** feature enabled, and be connected to a keyset equipped with a **PCDPM**.

Axessory Connect: DDE Capabilities

1 Standard DDE Capabilities

The DDE Interface on the **Axessory Connect** product is a programmable interface that allows straight-forward, yet powerful, integrations with MicroSoft Windows applications. The details for using the 'Standard' DDE capabilities are well documented in the **Axessory Connect** User's Guide so they are not included here.

2 Advanced DDE Capabilities

Some of the capabilities of the DDE Interface on the **Axessory Connect** are more advanced and are NOT documented in the User's Guide. These capabilities are documented in the following sections as they are intended for use by the OAI Developers.

2.1 Copy Protection

While some software packages implement copy protection via physical media (e.g., specially formatted diskettes which are difficult to copy or must be left in the machine while the software is running) or hardware devices (e.g., the notorious "dongle" which attaches to the parallel port of the user's PC), **Axessory Connect** provides a Log Serial Number feature via DDE for 3rd-party applications. This feature helps the an application developer implement a form of copy protection potentially without the added expense of a hardware device.

Note that these copy protection capabilities are not in any way guaranteed to be foolproof or invincible and are just provided as a service to help 3rd-party applications.

The copy protection capability for third-party software is provided through a DDE Request Transaction to log a serial number and a corresponding request to "un-log" the serial number. The DDE Service Name for the "LogSerialNumber" and "UnlogSerialNumber" requests is "AXXconnect" and the DDE Topic for the transaction is "System".

2.1.1 LogSerialNumber Request

This request will determine whether the supplied serial number is in use by any other device in the system. If it is, the request will return the FAIL response and the extension number of the device which is using the serial number. If the serial number is not in use, the request will return the PASS response and the extension number of the device with which **Axessory Connect** is associated. In either case, operation of the **Axessory Connect** will continue unaffected. When the **Axessory Connect** application is terminated, the re-log timer expires (see below), or the "Unlog" request is used (see below), the serial number will no longer be associated with the device.

Request format:

[LogSerialNumber ("Application|Topic", "Serial Number")]

Application|Topic An alphanumeric string which specifies what application to to what topic the DDE response should be returned .

Serial Number An alphanumeric string of up to 8 characters which specifies a unique serial number. The format of this serial number is *VPSSSSSS* where *V* is a single-character vendor code (defined by Inter-tel), *P* is a single-character product code (defined by the vendor) and *SSSSSS* is a six-character serial number which uniquely identifies the individual copy of the software. Because each of these characters can be a digit 0-9 or an alphabetic character A-Z, there are over 2 billion unique serial numbers for each vendor / product combination. Any lower-case alphabetic characters in the password string will be converted to upper-case.

Response format:

One of the following three responses will be sent back within a few seconds to inform the application of the result.

[LogSerialPass ("Extension Number ", "Serial Number ")]

[LogSerialFail ("Extension Number ", "Serial Number ")]

[LogSerialBusy ("Serial Number ")]

Extension Number This field will contain the extension number of the station with which the given serial number is associated; if the request was successful, this will be the extension of the station associated with the copy of **Axxessory Connect** software making the request, otherwise it will be the extension number of another station that has the serial number logged. By supplying this number, we allow the calling application to produce more informative error messages which include the identity of the other station where the serial number is logged.

Serial Number This field will contain the exact same string that was sent in the request.

Example:

A third party application attached to extension 1243 sends the following request to the PC Phone:

[LogSerialNumber ("MYAPP|SYSTEM", "XP568792")]

If the request is successful (i.e. no other extension has logged serial # XP568792), the response would be:

[LogSerialPass ("1243", "XP568792")]

If extension 1236 had already logged the serial number, the response would be:

[LogSerialFail ("1236", "XP568792")]

If another application is in the process of logging a serial number, this feature will be busy and you will need to try again later (typically after only a few seconds):

[LogSerialBusy ("XP568792")]

2.1.2 UnlogSerialNumber Request

This request is provided for those third-party applications which want to enforce software copy protection at run-time. The UnlogSerialNumber request 'release' the serial number so that other extensions can use it. It is a good practice for applications to always send this request whenever they are exiting. It is also a good practice to send request upon powerup (and before the "LogSerialNumber" request) in case the serial number was still 'logged in' from a previous use.

Request format:

[UnlogSerialNumber ("*Serial Number*")]

<i>Serial Number</i>	An alphanumeric string of up to 8 characters which specifies a unique serial number.
----------------------	--

Response format:

No response will be sent.

Example:

A third party application attached to extension 1243 sends the following request to the PC Phone:

[UnlogSerialNumber ("XP568792")]

2.2 Exceptions

2.2.1 Logging a serial number that is too long

If a third-party application attempts to log a serial number that is longer than eight characters, a "Fail" response will be returned with an extension number of 0.

2.3 Advanced OAI Commands

On some occasions there may be cases where an application would like to perform desktop OAI commands that are NOT provided in the set of 'standard' DDE capabilities of **Axxessory Connect**. This can be accomplished using a new (only available on **Axxessory Connect V2.0** or later) advanced DDE command explained in this section. Since this command can perform any desktop OAI command available on the system, it is only recommended for use by technical personnel who fully understand the desktop OAI commands and the implications of their use.

2.3.1 Direct OAI Command

This DDE command provides a mean to access any of the desktop OAI commands. Extreme care should be exercised when using this capability as sending certain OAI commands can have undesirable side affects on the operation of **Axxessory Connect**. The following OAI commands should normally be avoided completely: Echo, Version, Reset, and Interrupt Event.

Request format:

[DoCommand ("*command*")]

<i>command</i>	An alphanumeric string containing the desktop OAI command and all of its parameters in the exact format that would be send if directly connected to the RS232 port of the OAI link. Please refer to the Desktop OAI protocol specification (or the Desktop OAI help file) for a complete listing of all the Desktop OAI commands.
----------------	---

Response format:

none

Examples:

An application wants to increase the volume level setting of the telephone to maximum by sending the following DDE message to Axxessory Connect:

[DoCommand ("L8")]

An application wants to mute the microphone on the telephone by sending the following DDE message to Axxessory Connect:

[DoCommand ("M3")]

2.4 Do-Not-Disturb State

Occasionally it is useful for an application program to know the 'Do-Not-Disturb' state of the attached keyset. With **Axxessory Connect V2.0** (or later), attached applications can monitor this 'Do-Not-Disturb' state by reading a file named "DND.TXT" in the same directory as **Axxessory Connect**. When **Axxessory Connect** is running, this file will contain the current DND message from the keyset as follows:

"DND line 1" <CR><LF>

"DND line 2" <CR><LF>

So for example if the DND Message is: "OUT TO LUNCH UNTIL 1:00PM" then the DND.TXT file will contain:

"OUT TO LUNCH" <CR><LF>

"UNTIL 1:00PM" <CR><LF>

If the phone is not in the 'Do-Not-Disturb' state, then the DND.TXT file will contain:

" " <CR><LF>

The text in the DND.TXT file will always be updated within one second after a change in the 'Do-Not-Disturb' state of the attached keyset.

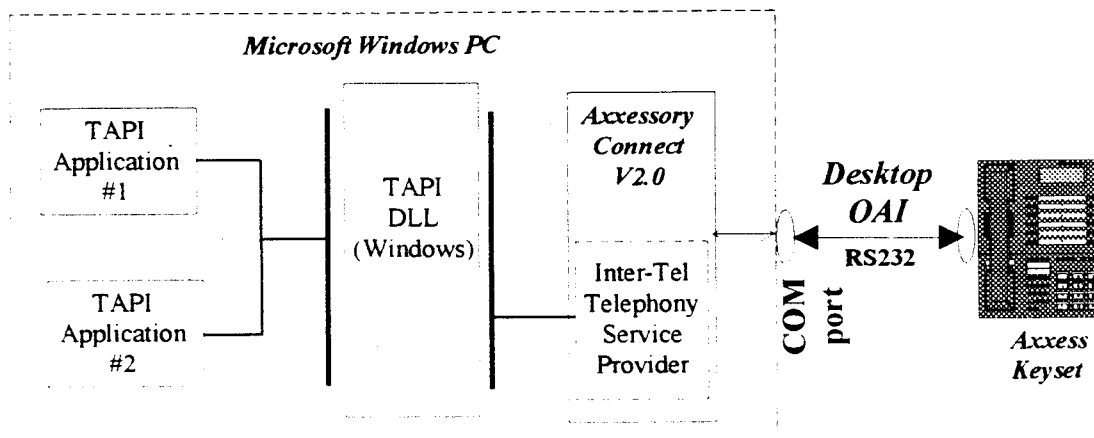
Axxessory Connect: TAPI Capabilities

1 Overview

Windows Telephony is a component of WOSA (Windows Open Services Architecture), and as such consists of an application programming interface (API) used by applications and a service provider interface (SPI) implemented by service providers.

With the WOSA model, applications can implement a single set of functions (using TAPI) that operate on many different telephone switches. The implementation details of these functions are handled by the telephone switch's service provider. Therefore, an application needs to know only the definition of the API, not its implementation.

The telephony model supplies a DLL component, called TAPI.DLL. This component resides between the Telephony API called by applications and the Telephony SPI implemented for the switch. To use the AXXESS switch as an example, TAPI-compliant applications call TAPI functions that are managed by TAPI.DLL and routed to the AXXESS service provider for execution. The relationships between these parts are shown in the following diagram:



The Telephony API and Inter-Tel AXXESS TSPI within the WOSA model

Inter-Tel's Telephony Service Provider Interface (TSPI), which is included with **Axxessory Connect V2.0** (or later), handles requests from applications by performing hardware specific operations on the AXXESS. Inter-Tel's TSPI communicates with the AXXESS through its desktop interface. That is, the AXXESS connects to the user's PC through an RS-232 serial interface to the user's keyset (using a PCDPM).

2 Inter-Tel's TSPI Model of the AXXESS

For the applications developer, using Telephony is using its two device classes: the line device class and the phone device class. Any physical telephonic device recognized by TAPI can be abstracted into one of these two classes, and can thus be treated in a consistent and reliable manner.

2.1 Line Device Model

The AXXESS TSPI models its interface as a single line device with 10 addresses, each of which can have at most one call appearance. The specific capabilities of the line device and each of the addresses are explained under the function descriptions for **lineGetDevCaps** and **lineGetAddressCaps**.

2.2 Phone Device Model

At present, the Inter-Tel AXXESS TSPI does not model the phone device.

3 Startup and Shutdown of TAPI

In order for an application to initialize with TAPI, certain line device functions must be called in order.

3.1 lineInitialize

An application establishes a TAPI connection by first calling the **lineInitialize** function. This function is not passed through to the TSPI but handled by the TAPI.DLL. Therefore, for complete details on how to use this function, please refer to the "Standard" TAPI specification.

One of the main purposes of **lineInitialize** is to establish a pointer to the application's callback function. After initialization, TAPI uses this pointer to notify the application of the asynchronous completion of functions and other events occurring on devices which the application has opened. Upon completion, the **lineInitialize** function returns two pieces of information to the application: an *application handle* and the number of available line devices. Inter-Tel's TSPI supports one line device.

3.2 lineNegotiateAPIVersion

The **lineNegotiateAPIVersion** function is used to negotiate the API version number to use. With this function call, the application provides the API version range it is compatible with. TAPI.DLL in turn negotiates with the line's service provider to determine which API version range it supports.

Since Inter-Tel's TSPI does not support extensions for its line device, **lineNegotiateAPIVersion** will return with zeros for its extension IDs. Again, please refer to the "Standard" TAPI specification for complete details on how to use this function.

3.4 lineOpen

To open a line device for any purpose—monitoring or control—the application calls the function **lineOpen**. This function opens the line (for a specified device ID), returning the service provider's handle for the device. The application can then use this handle to answer inbound calls, make outbound calls, or monitor call activities on the line for logging purposes.

Notice that an application should query the capabilities of a line, (using **lineGetDevCaps** described in the following section "Receiving Information"), before opening the line device. Upon calling **lineOpen**, the line will be opened (which includes allocating the serial port and initializing the serial interface), and a **LINE_LINEDEVSTATE** message will be generated with **LINEDEVSTATE_INSERTSERVICE** as the device state.

3.4 lineSetStatusMessages

This operation allows an application to specify which notification messages the service provider should generate for events related to the status changes for the specified line or any of its addresses.

3.5 lineClose

This function closes the specified open line device after aborting all outstanding calls and asynchronous operations on the device. Notice that an application can also call **lineShutDown** which deletes the call handles for any active calls and performs the equivalent of a **lineClose** on each open line.

4 Receiving Information

An application receives information in two ways: solicited and unsolicited. Solicited information is requested by the application through a function call such as **lineGetDevCaps** or **lineGetAddressCaps**. Unsolicited information arrives in the form of messages—most importantly call-state messages. Often, the two mechanisms are used together, as when an application receives a **LINE_CALLSTATE** message, after which it checks the information contained in the **LINECALLINFO** structure by calling **lineGetCallInfo**.

Inter-Tel's AXCESS service provider supports the following functions used to retrieve device capabilities, address capabilities, address IDs, address status, call information, call status, and line device status:

4.1 lineGetDevCaps

This function queries a specified line device to determine its telephony capabilities. It is important for an application to call this function before opening the line device. That is, the application must use **lineGetDevCaps** to decide which of (possibly) several line devices to use, depending upon the capabilities of each one.

Assuming a parameter valid **dwDeviceID** is supplied, the **LINEDEVCAPS** structure will be filled in as follows:

<i>Field Name</i>	<i>Setting</i>
dwProviderInfoSize dwProviderInfoOffset	The memory area referenced by these fields will contain the NULL-terminated string describing the service provider. For this release, the string is "Inter-Tel AXXESS SP"
dwPermanentLineID	This field will contain the permanent device ID, as found in TELEPHON.INI.
dwLineNameSize dwLineNameOffset	The memory area referenced by these fields will contain a NULL-terminated string containing the system-provided description for the station device to which the PC running the service provider is attached (e.g., John Smith).
dwStringFormat	STRINGFORMAT_ASCII.
dwAddressModes	LINEADDRESSMODE_ADDRESSID
dwNumAddresses	10
dwBearerModes	LINEBEARERMODE_VOICE
dwMaxRate	3100 (this isn't really used, but is provided for consistency).
dwMediaModes	LINEMEDIAMODE_INTERACTIVEVOICE
dwDevCapFlags	0
dwMaxNumActiveCalls	1
dwAnswerMode	LINEANSWERMODE_DROP LINEANSWERMODE_HOLD, reflecting the possibility that either could happen, based upon the programming of the phone system.
dwRingModes	1
dwLineStates	LINEDEVSTATE_INSERVICE LINEDEVSTATE_OUTOFSERVICE (future versions will report additional line states).
MinDialParams MaxDialParams DefaultDialParams	These will all indicate a dial speed, duration, and dial pause of 50, indicating 50 milliseconds. In reality, these values are programmed on the phone system and are not modifiable at the device level. The "wait-for-dialtone" value is set to 0 in each case, indicating that this functionality is not available.

4.2 lineGetAddressCaps

This function queries the specified address on the specified line device to determine its telephony capabilities. We return the same capabilities, regardless of the **dwAddressID** parameter, so long as that parameter is in the range 0-9. The **LINEADDRESSCAPS** fields filled in and their settings are as follows:

<i>Field Name</i>	<i>Setting</i>
dwLineDeviceID	The device ID of the line with which the address is associated (always 0).
dwAddressSize dwAddressOffset	A NULL-terminated string reflecting the extension number of the station Device to which the PC running the service provider is attached.
dwAddressSharing	LINEADDRESSSHARING_PRIVATE
dwAddressStates	LINEADDRESSSTATE_OTHER LINEADDRESSSTATE_INUSEZERO LINEADDRESSSTATE_INUSEONE LINEADDRESSSTATE_NUMCALLS
dwCallInfoStates	LINECALLINFOSTATE_OTHER LINECALLINFOSTATE_APPSPECIFIC LINECALLINFOSTATE_NUMOWNERINCR LINECALLINFOSTATE_NUMOWNERDECR LINECALLINFOSTATE_NUMMONITORS LINECALLINFOSTATE_DIALPARAMS LINECALLINFOSTATE_MONITORMODES LINECALLINFOSTATE_CALLERID LINECALLINFOSTATE_CALLEDID LINECALLINFOSTATE_CONNECTEDID LINECALLINFOSTATE_DISPLAY
dwCallerIDFlags	LINECALLPARTYID_ADDRESS LINECALLPARTYID_NAME
dwCalledIDFlags	LINECALLPARTYID_ADDRESS LINECALLPARTYID_NAME
dwConnectedIDFlags	LINECALLPARTYID_ADDRESS LINECALLPARTYID_NAME
dwRedirectionIDFlags	LINECALLPARTYID_UNAVAIL *
dwRedirectingIDFlags	LINECALLPARTYID_UNAVAIL *
dwCallStates	LINECALLSTATE_IDLE LINECALLSTATE_OFFERING LINECALLSTATE_DIALTONE LINECALLSTATE_ONHOLD LINECALLSTATE_DIALING LINECALLSTATE_BUSY LINECALLSTATE_CONNECTED LINECALLSTATE_PROCEEDING LINECALLSTATE_UNKNOWN
dwDialToneModes	LINEDIALTONEMODE_NORMAL
dwBusyModes	LINEBUSYMODE_UNAVAIL
dwSpecialInfo	LINESPECIALINFO_UNAVAIL
dwDisconnectModes	LINEDISCONNECTMODE_UNKNOWN LINEDISCONNECTMODE_UNAVAIL LINEDISCONNECTMODE_REJECT
dwMaxNumActiveCalls	1
dwMaxNumOnHoldCalls	1
dwAddrCapFlags	LINEADDRCAPFLAGS_DIALED LINEADDRCAPFLAGS_PARTIALDIAL
dwCallFeatures	LINECALLFEATURE_ANSWER LINECALLFEATURE_DIAL LINECALLFEATURE_DROP LINECALLFEATURE_GENERATEDIGITS LINECALLFEATURE_HOLD LINECALLFEATURE_UNHOLD LINECALLFEATURE_REDIRECT

* - May change in future versions

4.3 lineGetAddressID

An application uses this function to map a phone number (i.e., and extension or address) assigned to the line device back to its **dwAddressID** that is returned in the line's device capabilities.

This function will return **LINEERR_INVALIDADDRESS** if the **lpsAddress** parameter is anything other than the extension number of the station device to which the PC running the service provider is attached. If the correct **lpsAddress** parameter is supplied, the **DWORD** pointed to by **lpdwAddressID** will be set to 0, as this is the first address ID that matches the address specified by the **lpsAddress** parameter.

4.4 lineGetAddressStatus

This function queries the specified address for its current status.

If the **dwAddressID** parameter is outside the valid range (i.e., 0 through 9) of address IDs, the function will return **LINEERR_INVALIDADDRESSID**. Otherwise, the fields of the **LINEADDRESSSTATUS** structure will be filled in as follows:

<i>Field Name</i>	<i>Setting</i>
dwNumInUse	1 if there is a call associated with the address, 0 otherwise.
dwNumActiveCalls	1 if there is a call in any state other than ONHOLD at the address, 0 otherwise.
dwNumOnHoldCalls	1 if there is a call holding at the address, 0 otherwise.
dwNumOnHoldPendCalls	1 if there is a call in either the ONHOLDPENDCONF state or the ONHOLDPENDTRANSFER state, 0 otherwise.
dwAddressFeatures	If there is no call associated with the given address, this field will contain the value LINEADDRFEATURE_MAKECALL . Otherwise, it will be 0.

4.5 lineGetCallInfo

This function returns detailed information about the specified call. Assuming a valid **hCall** parameter is supplied, the **LINECALLINFO** structure will be filled in as follows:

<i>Field Name</i>	<i>Setting</i>
dwLineDeviceID	Always 0, as there's only one line.
dwAddressID	The address ID (0-9) with which the call is associated.
dwBearerMode	LINEBEARERMODE_VOICE
dwMediaMode	LINEMEDIAMODE_INTERACTIVEVOICE
dwAppSpecific	Retains the value passed in lineSetAppSpecific .
dwCallParamFlags	If the call was initiated with lineMakeCall , this field will be the same as the dwCallParamFlags field passed into that function via the LINECALLPARAMS structure. Otherwise, it will be 0.
dwCallStates	LINECALLSTATE_IDLE LINECALLSTATE_OFFERING LINECALLSTATE_DIALTONE LINECALLSTATE_ONHOLD LINECALLSTATE_DIALING LINECALLSTATE_BUSY LINECALLSTATE_CONNECTED LINECALLSTATE_PROCEEDING LINECALLSTATE_UNKNOWN LINECALLSTATE_DISCONNECTED
dwOrigin	either LINECALLORIGIN_UNKNOWN or some combination of LINECALLORIGIN_OUTBOUND and LINECALLORIGIN_INTERNAL or LINECALLORIGIN_EXTERNAL . This will never be set to LINECALLORIGIN_UNAVAIL .
dwReason	LINECALLREASON_UNAVAIL *
dwCountryCode	If the call was made using lineDial or lineMakeCall , this field will contain the country code passed in from the function call. Otherwise it will contain 0.
dwTrunk	Always 0xFFFFFFFF (Unknown).
dwCallerIDFlags	Either LINECALLPARTYID_UNKNOWN or some combination of LINECALLPARTYID_ADDRESS and LINECALLPARTYID_NAME , depending upon how much is known about the call.
dwCallerIDSize dwCallerIDOffset	If LINECALLPARTYID_ADDRESS is set in the dwCallerIDFlags field, these fields will contain the size and the offset of the NULL-terminated string specifying the address of the originator of the call. If the call is outbound, this will be the extension number of the station device to which the PC running the service provider is attached. If the call is inbound, this will be the CLASS-provided caller ID. This field can also be provided by an external entity manually modifying the information.
dwCallerIDSize dwCallerIDNameOffset	If LINECALLPARTYID_NAME is set in the dwCallerIDFlags field, these fields will contain the size and the offset of the NULL-terminated string specifying the name of the originator of the call. If the call is outbound, this will be the system-provided description (e.g., John Smith) of the station device to which the PC running the service provider is attached. If the call is inbound, this will be the CLASS-provided caller ID Name. This field can also be provided by an external entity manually modifying the information (e.g., via the szCalledParty parameter of lineRequestMakeCall).
dwCalledIDFlags	Either LINECALLPARTYID_UNKNOWN or some combination of LINECALLPARTYID_ADDRESS and LINECALLPARTYID_NAME , depending upon how much is known about the call.
dwCalledIDSize dwCalledIDOffset	If LINECALLPARTYID_ADDRESS is set in the dwCalledIDFlags field, these fields will contain the size and the offset of the NULL-

	terminated string specifying the address of the originally addressed party of the call. If the call is outbound, this will be the dialable address of the called party. If the call is inbound, this will be either the extension number of the station device to which the PC running the service provider is attached (if the call is an internal one) or the DNIS- or DISA-provided address dialed by the outside caller (if the call is an external one and the information is available).
dwCalledIDNameSize dwCalledIDNameOffset	If LINECALLPARTYID_NAME is set in the dwCalledIDFlags field, these fields will contain the size and the offset of the NULL-terminated string specifying the name of the originally addressed party of the call. If the call is outbound, this field will be empty, unless supplied to the AXXESS system by an external entity (e.g., via the szCalledParty parameter of lineRequestMakeCall). If the call is inbound, this will be either the system-provided description (e.g., John Smith) of the station device to which the PC running the service provider is attached (if the call is an internal one) or the system-provided description of the trunk device to which the call rang in (if the call is an external one and the information is available).
dwConnectedIDFlags	Either LINECALLPARTYID_UNKNOWN or some combination of LINECALLPARTYID_ADDRESS and LINECALLPARTYID_NAME, depending upon how much is known about the call.
dwConnectedIDSize dwConnectedIDOffset	If LINECALLPARTYID_ADDRESS is set in the dwConnectedIDFlags field, these fields will contain the size and the offset of the NULL-terminated string specifying the address of the other party of the call. If the call is outbound, these will contain the same string as that referenced by dwCalledIDSize and dwCalledIDOffset (since this service provider does not currently handle redirection information). If the call is inbound, these fields will contain the same string as that referenced by dwCallerIDSize and dwCallerIDOffset .
dwConnectedIDNameSize dwConnectedIDNameOffset	If LINECALLPARTYID_NAME is set in the dwConnectedIDFlags field, these fields will contain the size and the offset of the NULL-terminated string specifying the name of the other party of the call. If the call is outbound, these will contain the same string as that referenced by dwCalledIDNameSize and dwCalledIDNameOffset (since this service provider does not currently handle redirection information). If the call is inbound, these fields will contain the same string as that referenced by dwCallerIDNameSize and dwCallerIDNameOffset .
dwDisplaySize dwDisplayOffset	If the call being referenced is the active (i.e., connected or highest priority ringing) call on the line, these fields will contain the contents of the two-line LCD display currently being shown on the station device to which the PC running the service provider is attached. Even if the station device does not have a two line LCD display, this string will still be valid.

* - May change in future versions

4.6 lineGetCallStatus

This function returns the current status of the specified call. Assuming a valid **hCall** parameter is supplied, the LINECALLSTATUS structure will be filled in as follows:

<i>Field Name</i>	<i>Setting</i>
dwCallState	One of LINECALLSTATE_IDLE, LINECALLSTATE_OFFERING, LINECALLSTATE_DIALTONE, LINECALLSTATE_ONHOLD, LINECALLSTATE_DIALING, LINECALLSTATE_BUSY, LINECALLSTATE_CONNECTED, LINECALLSTATE_PROCEEDING, LINECALLSTATE_UNKNOWN, or LINECALLSTATE_DISCONNECTED, reflecting the actual state of the call.
dwCallStateMode	If dwCallState is LINECALLSTATE_DIALTONE, this will always be LINEDIALTONEMODE_NORMAL. If dwCallState is LINECALLSTATE_DISCONNECTED, this will be either LINEDISCONNECTMODE_UNKNOWN (for a normal disconnect) or LINEDISCONNECTMODE_REJECT (if the call was disconnected because the called party was in Do-Not-Disturb). If dwCallState is LINECALLSTATE_BUSY, this will be LINEBUSYMODE_STATION.
dwCallFeatures	This field will indicate all the features that are allowable, given the current state of the call. Refer to the description of the appropriate feature for a list of the states in which it is allowed.

4.7 lineGetID

This function returns a device ID for the specified device class associated with the selected line, address, or call.

If **lpszDeviceClass** is anything other than "tapi/line", this function will return LINEERR_NODEVICE. If **dwSelect** is LINECALLSELECT_ADDRESS and **dwAddressID** is not a valid address (i.e., 0-9), the function will return LINEERR_INVALIDADDRESSID. If **dwSelect** is LINECALLSELECT_CALL and **hCall** is not a valid call handle, the function will return LINEERR_INVALIDCALLHANDLE. Otherwise, the VARSTRING pointed to by **lpDeviceID** will be filled in as follows:

<i>Field Name</i>	<i>Setting</i>
dwStringFormat	STRINGFORMAT_BINARY
dwStringSize	4
dwStringOffset	The memory area referenced by this field will contain a DWORD containing the ID of the line device associated with the specified line handle, call handle, or address ID. Since there is only one line, this will always be 0.

4.8 lineGetLineDevStatus

This operation queries the specified open line device for its current status. Assuming **hLine** is valid, the LINEDEVSTATUS structure will be filled in as follows upon return:

<i>Field Name</i>	<i>Setting</i>
dwNumActiveCalls	The number of calls ringing on the line and / or connected.
dwNumOnHoldCalls	The number of calls that are on hold.
dwNumOnHoldPendCalls	The number of calls that are in the ONHOLDPENDCONF state or the ONHOLDPENDTRANSFER state.
dwLineFeatures	LINEFEATURE_MAKECALL, if there are no active calls, 0 otherwise.
dwRoamMode	LINEROAMMODE_UNAVAIL.
dwDevStatusFlags	LINEDEVSTATUSFLAGS_CONNECTED LINEDEVSTATUSFLAGS_INSERVICE

5 TAPI Functions - Basic Call Control

This section describes the TAPI functions used to support basic call control capabilities on Inter-Tel's AXXESS service provider.

5.1 lineMakeCall

This function places a call on the specified line to the specified destination address.

If the line is currently busy (i.e., there is an actively ringing or connected call on the line), the function will return `LINEERR_RESOURCEUNAVAIL`. If a bearer mode other than `LINEBEARERMODE_VOICE` is specified in the call parameters, `LINEERR_INVALIDBEARERMODE` will be returned. If a media mode other than `LINEMEDIAMODE_INTERACTIVEVOICE` is specified, `LINEERR_INVALIDMEDIAMODE` will be returned. If call parameter flags other than `LINECALLPARAMFLAGS_IDLE` and `LINECALLPARAMFLAGS_ORIGOFFHOOK` are specified, `LINEERR_INVALIDCALLPARAMS` will be returned. The call parameters are otherwise ignored, but the relevant values are stored with the call for later retrieval.

If no destination address is specified, a new call will be created and the line will transition to the `LINECALLSTATE_DIALTONE` state. The `lineDial` function (see below) can then be used at this point to dial the destination address.

If a destination address is specified, how it is dialed depends upon the access code and minimum length values programmed via `lineConfigDialog`:

- If the length of the destination address is greater than or equal to the minimum outside number length, the outside line access code will be prepended to the number before it is dialed.
- If the length of the destination address is less than the minimum outside number length, the number will be dialed as is.

Note that, as mentioned below under `lineDial`, length checking is in effect only for number dialed using `lineMakeCall`.

5.2 lineDial

This function dials the specified dialable number on the specified call.

Unlike `lineMakeCall` (see above), the AXXESS service provider will not attempt to differentiate between internal and external destination when `lineDial` is used; the digits are dialed as-is. This feature is only allowed for calls in the *dialtone* or *dialing* states. The country code will be ignored, but will be reflected in any calls to `lineGetCallInfo`.

5.3 lineAnswer

This function answers the specified *offering* call. The **lpsUserUserInfo** and **dwSize** parameters are ignored. This feature is only allowed for calls in the *offering* state.

5.4 lineAccept

This function accepts the specified *offering* call. The **lpsUserUserInfo** and **dwSize** parameters are ignored. This feature is only allowed for calls in the *offering* state.

5.5 lineDrop

This function drops or disconnects the specified call. When this function returns, the call will be in the *idle* state.

As with **lineAnswer**, the **lpsUserUserInfo** and **dwSize** parameters are ignored. Because the AXCESS service provider can drop calls only to which it is connected, this feature is allowed for calls in the CONNECTED state only. In addition, the service provider keeps track of whether a given call is "owned" by a TAPI application, in order to avoid dropping calls initiated or answered by other applications. In order for a call to be considered "owned" by a TAPI application, it must be either initiated, answered, or placed on hold by a TAPI application.

This operation is ignored (i.e., no error condition is indicated, but no action is performed) if the call is in the *idle* state.

5.6 lineGenerateDigits

This function initiates the generation of the specified digits on the specified call using DTMF tones for digital signaling.

The AXCESS service provider is only capable of generating DTMF digits; a value of **LINEDIGITMODE_PULSE** in the **dwDigitMode** parameter will result in an error condition with a return value of **LINEERR_INVALIDDIGITMODE**. The duration of digits on the AXCESS system is programmed at the system level, therefore the **dwDuration** parameter is ignored. This feature is allowed if the call is in the **DIALTONE**, **DIALING**, **PROCEEDING**, or **CONNECTED** states.

The **LINEGENERATETERM_DONE** message will be generated almost instantaneously; the reason for this is that the service provider isn't notified when digit generation is complete and, consequently, it has no means of interrupting the process once it has begun. For this reason, it is next to impossible to ever cause the service provider to generate a **LINEGENERATETERM_CANCEL** message; this would be indicative of some sort of internal overflow condition.

5.7 lineHold

This function places the specified call on hold.

This feature is only valid in the LINECALLSTATE_CONNECTED or _PROCEEDING states. Note that a TAPI application can place a call on hold even if it did not initiate or answer that call.

5.8 lineUnhold

This function retrieves the specified held call.

This operation is valid only if the call is in the LINECALLSTATE_ONHOLD state. If this is the case, the call will be returned to active / connected status. Otherwise, the service provider will return LINEERR_INVALIDCALLSTATE.

5.9 lineRedirect

This function redirects the specified *offering* call to the specified destination address. That is, this function deflects the *offering* call to another address and reverts it to the *idle* state.

The **lpszAddress** is subject to the same translation rules as the **lpszAddress** parameter of **lineMakeCall** (see above). The country code will be ignored.

6 TAPI Functions - Transferring Calls

This section describes the supplementary TAPI functions used to support the transferring of calls. TAPI provides two mechanisms for call transfer: blind transfer and consultation transfer.

- In blind transfer (or single-step transfer), an existing call is transferred to a specified destination address in one phase using **lineBlindTransfer**.
- In a consultation transfer, the existing call is first prepared for transfer to another address using **lineSetupTransfer**. This places the existing call on consultation hold, and identifies the call as the target for the next transfer-completion request. **lineSetupTransfer** also allocates a consultation call that can be used to establish the consultation call with the party to which the call will be transferred. The application can dial the extension of the destination party on the consultation call (using **lineDial**), or it can drop and deallocate the consultation call and instead activate an existing held call (using **lineUnhold**). Finally, the application completes the transfer of the call on transfer hold to the destination party by using **lineCompleteTransfer**. At this point, both call appearances revert to the *idle* state.

6.1 lineBlindTransfer

This function performs a single-step transfer of the specified call to the specified destination address. Blind transfer is different than a consultation transfer in that no consultation call is made visible to TAPI.DLL. (See **lineSetupTransfer** for implementing a consultation transfer, below.)

The **IpszDestAddress** will be interpreted in the same manner as the **IpszDestAddress** parameter of the **lineMakeCall** function (see above).

6.2 lineSetupTransfer

This function initiates the transfer of the call specified by **hdCall**. It establishes a consultation call, **IphdConsultCall**, on which the party can be dialed that can become the destination of the transfer.

The call parameters are treated the same as those of **lineMakeCall** (see above). While the consultation call exists, the original call transitions to the ONHOLDPENDTRANSFER state.

6.3 lineCompleteTransfer

This function completes the transfer of the specified call to the party connected in the consultation call.

The **hConsultCall** parameter must be a handle that was previously returned by **lineSetupTransfer**. The only transfer mode supported is LINETRANSFERMODE_TRANSFER, so the **IphConfCall** parameter is ignored. Therefore, an application must use the "standard" TAPI conference functions in order to perform conferencing.

7 TAPI Functions - Conferencing

This section describes the supplementary TAPI functions used to support call conferencing.

A conference call must begin as a regular two-party call, such as a call established with **lineMakeCall**. Once the two-party call exists, additional parties can be added, one at a time. Calling **lineSetupConference** prepares a given call for the addition of another party, and this action establishes the conference call. This operation takes the original two-party call as input, allocates a conference call, connects the original call to the conference, and allocates a consultation call whose handle is returned to the application.

The application can then use **lineDial** on the consultation call to establish a connection to the next party to be added. **lineDrop** can be used to abandon this call attempt. The third party is added with the function **lineAddToConference**, which specifies both the conference call and the consultation call.

To add additional parties to an existing conference call, the application uses **linePrepareAddToConference**. When calling this function, the application supplies the handle of an existing conference call. The function allocates a consultation call that can later be added to the conference call and returns a consultation call handle to the application. This conference call is then placed on conference hold. Once the consultation call exists, it can be added to the existing conference call with **lineAddToConference**.

Once a call becomes a member of a conference call, the member's call state reverts to *conferenced* while the conference call's state becomes *connected*. The call handle to the conference call and all the added parties remain valid as individual calls.

The AXXESS service provider is limited somewhat by the way LINE_CALLSTATE events can be received about calls in a conference. For example, if one of the members disconnects by hanging up, a call-state message will be reported for the disconnecting party only if a three-party conference is reverting to a two-party call. That is, for conference calls with more than three connected parties, call-state message will **not** be reported for a disconnecting party. *This may be fixed in future versions of AXXESS and AXXENT software.*

Also, notice that our service provider does not support the **lineRemoveFromConference** function since our desktop interface does not support third-party call control.

7.1 lineSetupConference

This function sets up a conference call for the addition of a third party. Notice that the AXXESS service provider requires a pre-existing two-party call in order to establish the conference.

This function allocates a consultation call for connecting to the party that is to be added to the call and returns a newly created conference call with a call state of ONHOLDPENDCONF.

The **hCall** parameter must be valid. The **hLine** parameter is ignored, and the **dwNumParties** parameters are ignored. The call parameters are treated the same as those of **lineMakeCall** (see above).

7.2 **lineAddToConference**

This function adds the call, specified by **hdConsultCall**, to the conference call specified by **hdConfCall**.

The **hConfCall** parameter must be a handle previously returned by **lineSetupConference**. The **hConsultCall** parameter must be a handle previously returned by **lineSetupConference** or **linePrepareAddToConference**. That is, the AXXESS Service Provider does not allow an arbitrary call to be added to the conference (i.e., a call that was setup using **lineMakeCall**).

7.3 **linePrepareAddToConference**

This function prepares an existing conference call for the addition of another party. It creates a new, temporary consultation call which can subsequently be added to the conference with the **lineAddToConference** function.

The **hConfCall** parameter must be a handle previously returned by **lineSetupConference**. The call parameters are treated the same as those of **lineMakeCall** (see above).

7.4 **lineRemoveFromConference**

The AXXESS service provider **cannot** support this function since Inter-Tel's Desktop Interface model does not support third-party call control. That is, the AXXESS service provider can perform functions on the controlling keyset only. Therefore, we cannot remove a conferenced call located on a third-party station.

Inter-Tel currently has no plans to support this function in future versions of the AXXESS software or the service provider.

8 TAPI Functions - Miscellaneous

This section describes miscellaneous TAPI functions supported by the AXXESS service provider that could not be categorized above. Also, the end of this section lists commands that the service provider currently does not support.

8.1 lineConfigDialog

This function displays a configuration dialog for the AXXESS line device. The configurable options are the minimum number of digits necessary for an outside number and the access code that should be used to obtain an outside line. These two parameters are used when a complete destination address is supplied with **lineMakeCall** (see above) to determine whether the destination is internal or external. Any changes made will take place immediately; the service provider allows changes to be made while the line is open.

8.2 lineSetAppSpecific

This operation sets the application-specific field of the specified call's LINECALLINFO structure. Its usage is entirely defined by the application and is uninterpreted by the Telephony API or the service provider.

The **dwAppSpecific** value is saved for later retrieval via **lineGetAppSpecific**. When this field is changed, the service provider sends a LINE_CALLINFO message with an indication that the **dwAppSpecific** field has changed.

8.3 Functions Not Supported

This section lists any TAPI functions the Inter-Tel AXXESS service provider currently does not support:

lineCompleteCall	linePickup
lineDevSpecific	lineRemoveFromConference
lineDevSpecificFeature	lineSecureCall
lineForward	lineSendUserUserInfo
lineGatherDigits	lineSetCallParams
lineGenerateTone	lineSetDevConfig
lineGetDevConfig	lineSetMediaControl
lineGetIcon	lineSetMediaMode
lineMonitorDigits	lineSetTerminal
lineMonitorMedia	lineSwapHold
lineMonitorTones	lineUncompleteCall
lineNegotiateExtVersion	lineUnpark
linePark	

9 TAPI Functions - Miscellaneous

This section indicates which messages the AXCESS TAPI Service Provider currently generates.

LINE_ADDRESSSTATE - Not currently generated

LINE_CALLINFO - See the **dwCallInfoStates** field under **lineGetAddressCaps** for an enumeration of the various LINECALLINFOSTATE values used.

LINE_CALLSTATE - See the **dwCallState** field under **lineGetCallStatus** for an enumeration of the various LINECALLSTATE values used.

LINE_LINEDEVSTATE - As mentioned under **lineGetDevCaps**, LINEDEVSTATE_INSERVICE and LINEDEVSTATE_OUTOFSERVICE are the only LINEDEVSTATE values currently generated.

LINE_REPLY - Messages are generated for all asynchronous requests.

LINE_GENERATE - As mentioned under **lineGenerateDigits**, this message is generated almost immediately, as the service provider is unaware of the actual point at which digit generation is complete.

SECTION 7: Sample Code

Often times it is very useful for software developers to be able to refer to example source code when developing an application program. Therefore, we've included the following:

1. **BIG DND program** - this program, written in Visual Basic 3.0, is an MS-Windows program that works with the *Axxessory Connect (V2.0)*. It pops up and displays the Do-Not-Disturb message whenever the attached keyset is placed into DND.

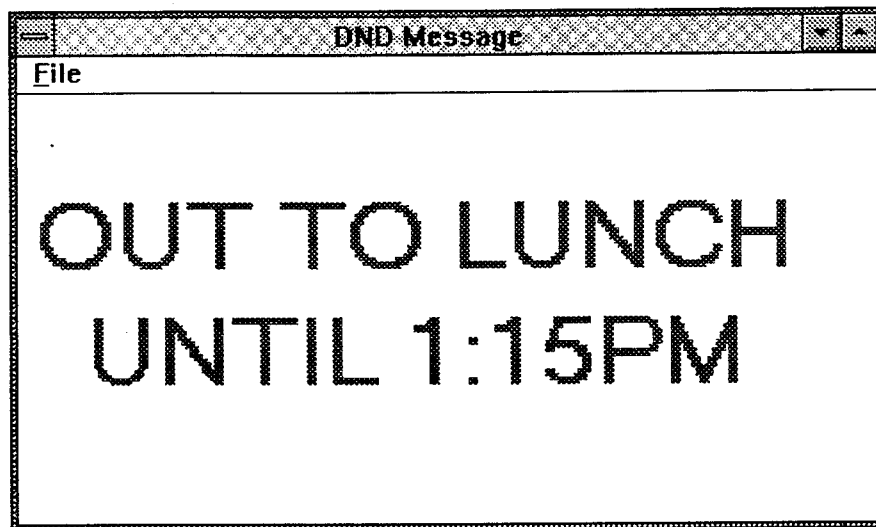
2. **FWD program (DDE Version)** - this program, written in Visual Basic 3.0, is an MS-Windows program that works with the *Axxessory Connect* using DDE links. With it you can set up 'Forwarding rules' on your PC screen that allow you to forward calls from specific people to specific destinations.

The source and executable files are included on the "*Desktop OAI Samples*" disk, and the "Application Notes" and "User's Guides" follow in this section.

Overview:

Occasionally when you walk to someone's office, you see they're not there, so you look at their Axxess (or Axxent) phone to see the DND message. But, if they have other messages waiting on their phone, you can't see their DND message. So, maybe they just stepped out for a minute or maybe they left for a meeting and they don't expect to return for 2 hours, but you don't know. We've experienced this little frustration, so we wrote a little CTI application to help out in this situation.

This sample CTI application, called *BIG DND*, monitors the Do-Not-Disturb state of the keyset, and whenever the keyset is put into DND, this program pops up, fills up the PC screen and displays the DND message on the screen in a large font. Thus you can see the person's DND message from way across the room similar to the following:



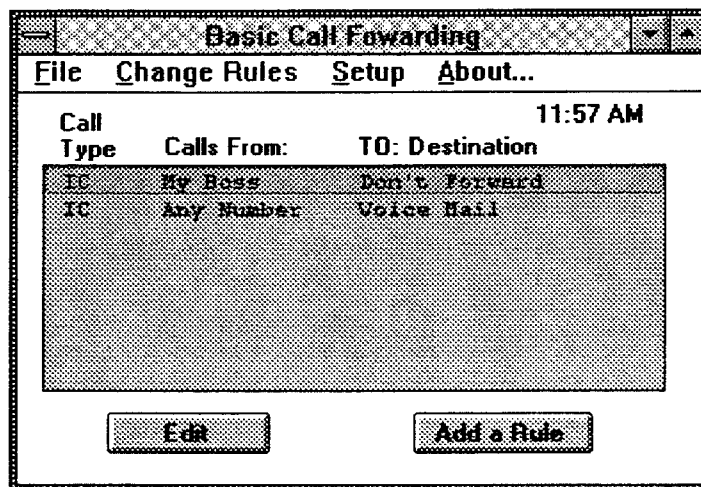
When the keyset is taken out of DND, this program automatically pops back down minimizing itself to an icon again.

This program is written in Visual Basic (3.0) and it must be installed in and run from the same directory as *Axxessory Connect (V2.0)* for it to work properly. The program simply monitors the DND.TXT file which *Axxessory Connect* keeps updated with the latest Do-Not-Disturb information from the keyset.

Overview:

Using the Visual Basic programming environment and the DDE link interface on the *Axxessory Connect*, it is fairly easy to develop powerful Windows telephone applications. This example shows a program called *BasicFWD* which provides enhanced telephone call forwarding. The user simply adds call forwarding rules that include what type of call, what numbers to forward from, and then where the calls should be forwarded. The *BasicFWD* program then monitors all calls to the (attached) phone, determines if/when one of the forwarding rules applies, and then appropriately sends a command to have the call forwarded to the desired destination.

The main screen of the *BasicFWD* program looks like:



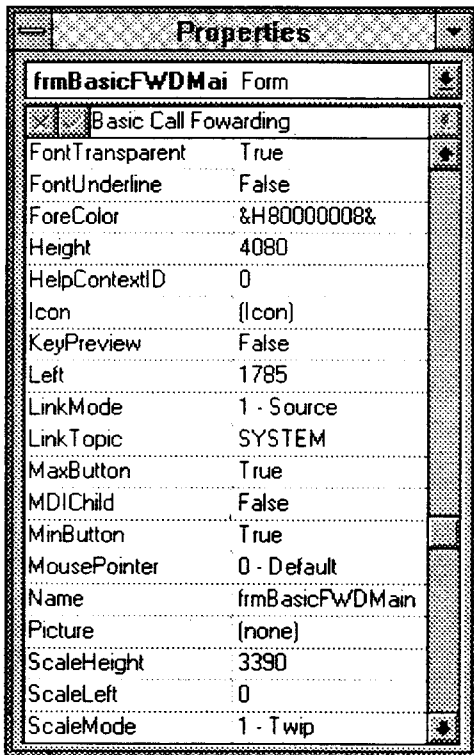
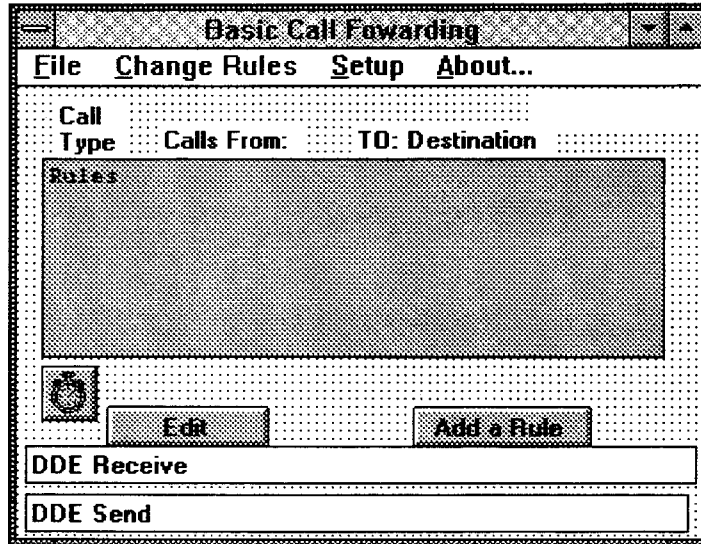
The gray window in the center of the screen shows the 'forwarding rules'. To edit one of these rules or add a new rule you can use the **C**hange Rules menu or the **E**dit or **A**dd a Rule button. Then the *Add/Edit a Forwarding Rule* screen will appear and you will be able to see and change the details of that call forwarding rule, as shown in the following screen:

All messaging to/from the *BasicFWD* program from/to the *Axcessory Connect* is accomplished using DDE commands. This includes messages from the *Axcessory Connect* whenever a call (1) starts ringing, (2) is answered, or (3) exits the phone, and messages from the *BasicFWD* program to forward (Divert) a call. From the main screen use Setup and Change DDE Destination to make sure the DDE link is properly programmed to send to the *Axcessory Connect* program as shown in the following screen:

Also, in the *Axcessory Connect* program, you must set up DDE links to the **BASICFWD** program with a topic of **SYSTEM** so that DDE messages will be sent when (1) **CallRinging**, (2) **CallCleared**, and (3) **CallConnected** events occur.

DESIGN:

The main form (**frmBasicFWDMain**) of the **BasicFWD** program is shown along with its attributes in the following screens. Note that **frmBasicFWDMain** is set up to receive System DDE messages (**LinkMode** = 1-Source and **LinkTopic** = System) so it will be the recipient of any DDE messages sent from the *Accessory Connect*.



When a DDE message (sent as a **LinkExecute**) is received by the *BasicFWD* program the Subroutine "Form_LinkExecute" is invoked. Upon receipt of a valid DDE message from the *Accessory Connect*, this routine simply adds (if "CallRinging") or deletes (if

"CallCleared" or "CallConnected") a call to/from its list of active calls (stored in a Call Queue). The code is shown as follows:

RECEIVING DDE EXECUTE COMMANDS from the AXCESSORY CONNECT:

```
Sub Form_LinkExecute (CmdStr As String, Cancel As Integer)
    Dim Cmd As Variant
    Dim CallID As Variant
    Dim FromNum As Variant
    Dim CallType As Integer
    Dim Success As Integer

    DDEText.Text = CmdStr
    Cmd = ParseDDE(CmdStr, CallID, FromNum, CallType)
    Select Case (LCase(Cmd))
        Case "callringing"
            Success = Add_Call_Q(CallID, FromNum, CallType)
        Case "callcleared"
            Del_Call_Q CallID
        Case "callconnected"
            Del_Call_Q CallID

    End Select
    Cancel = False
End Sub
```

The BasicFWD program then uses a Timer (called Timer1) to periodically (once every second) check each call in the call queue against each of the 'Call Forwarding Rules'. If it determines that is it time to 'forward' one of the calls, it then sends a DDE message ("DivertCall ... ") to the *Axxessory Connect*. Sending a DDE message is accomplished through the "DDEsend" textbox (properties shown below) with the code in the **Timer1_Timer** subroutine which is shown below:

SENDING DDE EXECUTE COMMANDS to the AXCESSORY CONNECT:

```
Sub Timer1_Timer ()
    .....
    .....
    'Send "Divert" DDE Message to PC Phone
    Dest = Trim(FWDrules(FRule).ToNum)
    If Strlen(Dest) > 4 Then 'Outside Call
        Trk = Trim(TrkGrp)
        DDEsend.Text = "[DivertCall (*** & CallQ(Index).CallID & ***,***
                        & Trk & ***,*** & Dest & ***,****)]"
    Else 'Intercom Call
        DDEsend.Text = "[DivertCall (*** & CallQ(Index).CallID & ***,***
                        & Dest & ***,****,****)]"
    End If
    DDEsend.LinkItem = "" 'DDEsend.Text
    DDEsend.LinkMode = 2 'Setup Manual DDE link
    DDEsend.LinkExecute DDEsend.Text
    DDEsend.LinkMode = 0 'Close DDE Link
    .....
End Sub
```

NOTE: The `DDEsend.LinkTopic` field is previously configured (by the user) from menu selection **Setup, Change DDE Destination**.

User's Guide: *AXXESS BasicFWD* Program

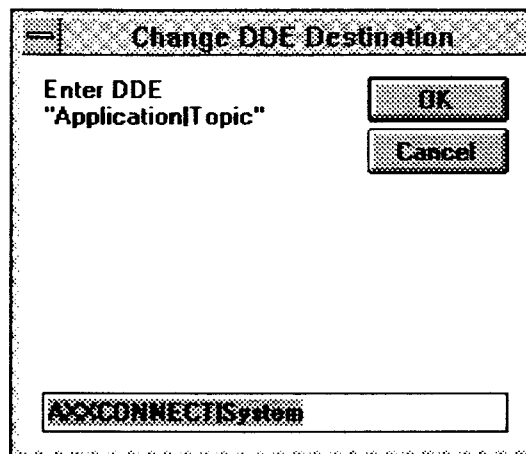
Overview:

This program, called *BasicFWD*, is an MS-Windows based program that works in conjunction with *Axxessory Connect* software to provide 'enhanced' call forwarding. With it, you can setup call forwarding for specific outside or inside numbers to specific inside or outside destinations.

Installing and Setting up *BasicFWD*:

The program can be installed using MS Windows program manager by selecting **R**un and the filename: **SETUP.EXE** from the installation floppy disk. Once installed under windows you can simply double-click on the *BasicFWD* Icon to run the program. It's a good idea to then add *BasicFWD* (and *Axxessory Connect*) to your "STARTUP" folder so these programs automatically run whenever you start up MS Windows.

The *BasicFWD* program communicates with *Axxessory Connect* using 'DDE links' which must be setup in both programs for them to work together. In the *BasicFWD* program, from the main screen on the main menu, select **S**etup and **C**hange DDE Destination and then set the Application|Topic to: **AXXCONNECT|SYSTEM** as shown in the following screen. And then select **F**ile and **S**ave from the main menu.



In *Axxessory Connect* you must program DDE links to *BASICFWD* for "Call Ringing", "Call Connected", and "Call Cleared" events. This is done simply by adding a "New Event" to each and putting setting the Application Name to: **BASICFWD** and the Topic to: **SYSTEM** as shown in the following screen.

Edit Call Ringing Event

Service Name:

Topic:

Executable Name: Run if necessary

Transaction Type: Event enabled

DDE Execute / Request String:

DDE Reply Format:

Audibly Ringing Calls Only

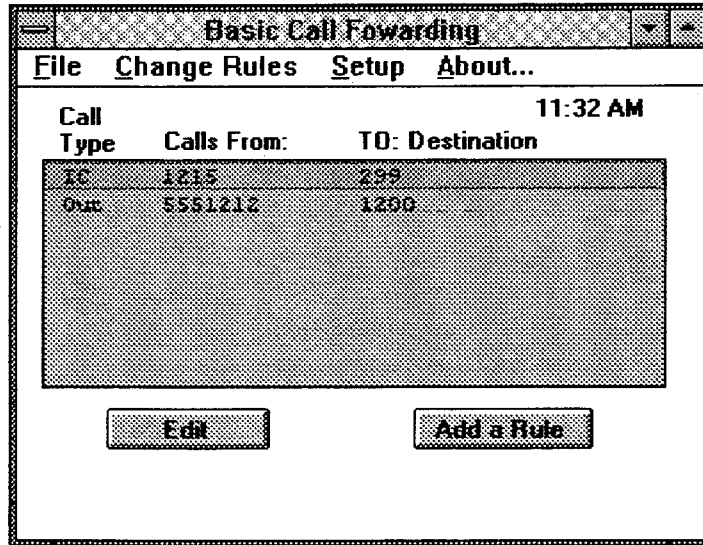
Call Types:

- Forward Hold Recall Queue Callback
- Transfer Transfer Recall

The DDE links should now be setup and ready to use.

Using BasicFWD:

The main screen, called "Basic Call Forwarding", is what you'll see first when you run the program. The gray window in the center of this screen shows the call forwarding rules that you have programmed. Through these call forwarding rules you decide which calls you want to forward and to where you want them forwarded.



Its a good practice to save (use File and Save) the BasicFWD configuration any time you make changes that you want to keep just in case an abnormal exit from Windows occurs.

Setting up Forwarding Rules:

You can edit the highlighted rule or add a new rule using either the **Edit** and **Add a Rule** buttons or from the **Change Rules** menu selections. When choose to add or edit a rule the "Add/Edit a Forwarding Rule" screen will appear similar to as follows:

The individual fields and buttons for this screen are:

- 1. Call Type** - In this field you decide what type of calls you want to forward: *Intercom*, *Outside*, or *All* (both).
- 2. Calls From** - Here you select if calls from a specific number or range of numbers are to be forwarded. If you leave this field blank it means "Calls from Any Number". In this field you can specify a specific intercom or outside number (like 9619000 or 1215) or use wild cards to specify a range of numbers (like 308+ for all calls from Nebraska or 121? for all extensions 1210 through 1219). You can also program the "Refer to As" field to make the rule more readable on the 'rules display' on the main screen. For example, for the number 308+ you might set "Refer to As" as "Nebraska".
- 3. FWD Destination** - Here you specify where you want the calls be forwarded (if at all). That is, you can set it to: an Intercom number, an Outside phone number, or leave it blank. Leaving it blank is a special case saying "*Don't Forward the call if this rule applies*", which is very useful for cases where you want certain calls to ring at your phone and not be forwarded - like the example of letting your bosses calls ring through to you. Phone numbers greater than 5 digits in length are considered to be 'outside numbers' in which case the 'Trunk Access Code' (see Other Features) is automatically used, so a trunk access code should **NOT** be included in this field. Again, like the 'Calls From' field, the "Refer to as" field allows you to describe the number to make it more readable on the main screen.
- 4. Enable Rule** - This checkbox must be set for the rule to be enabled. This provides a convenient way to disable a rule but not delete it so you can keep in around to be used later.

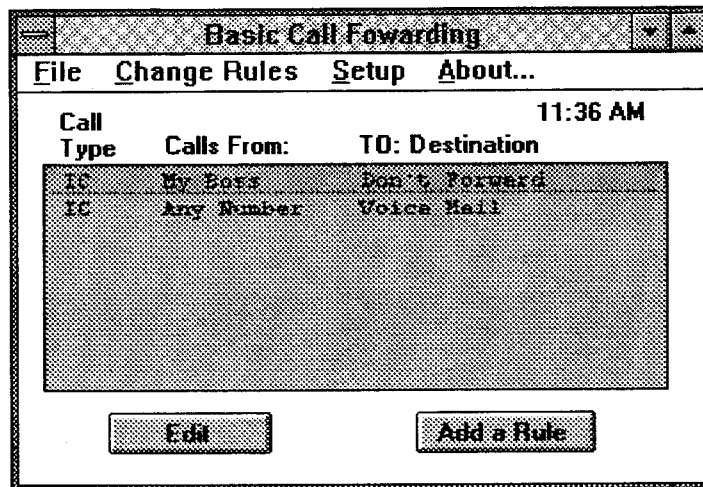
5. **S**ave - This button saves the changes you've made to the rule and returns to the main screen.

6. **C**ancel - This button avoids saving any changes you may have made to the rule and returns to the main screen.

From the **C**hange Rules menu selections you can also **D**elete an individual rule or **C**lear All (delete all) of the forwarding rules.

Receiving/Forwarding Calls:

An important aspect of the BasicFWD program to remember is that the call forwarding rules apply in the order that they appear on the screen. That is, when a call starts ringing, the program will check the first rule, then the 2nd rule, and so forth until it finds a rule that applies, so the order the rules appear on the screen can be important. In the example where you want to forward all your calls to Voice Mail except calls from your boss, the rules must be in the specific order where the rule from "My Boss" appears before the other rule like:



You can change the order of the forwarding rules on the main screen using the 'Drag & Drop' method. That is, click and hold the left mouse button on the rule you want to move and then simply drag it to its new location and release the left mouse button.

Other Features:

Trunk Access - Here you can select the trunk access code you want to use when calls are forwarded to an outside number. If you program this 'blank' then *Axxessory Connect* will use its own trunk access code field.

DDE Debug Windows - Sometimes when your trying to troubleshoot DDE problems it is helpful to be able to 'see' the DDE events. By turning on these windows you will be able to see the last DDE event send from and received by the *BasicFWD* program.

Call Queue - Sometimes it is useful to see some specifics on calls that are actively ringing and this pop-up window provides that information.

