LDK TSP

User’s Guide

Issue 3.0Ac

LG Electronics, Inc.

REVISION HISTORY

Version	Date	Description of Change	S/W Version
Issue 1.0	MAR/12/2001	Initial Release	1.0Ae
Issue 2.0Aa	FEB/05/2002	Add Windows 2000 Server	2.0Aa
Issue 3.0Aa	MAY/31/2004	Add IP LDK Features Add Windows Server 2003	3.0Aa
Issue 3.0Ab	AUG/18/2004	Add Hot desk features	3.0An

CONTENTS

About This Manual

This manual describes the use of LG Electronics, Inc (LGE)' s LDK TSP, and the TSP (Telephony Service Provider) supporting Microsoft TAPI 2.1 for telephony applications. Those applications are referenced to as the TAPI application in the reminder of this manual.

Audience

This manual is for anyone using LDK TSP to develop TAPI applications.

Associated Documentation

LDK Series Documentation

- Programming Manual for Digital Key Telephone System
- User's Guide for Digital Key Telephone

TAPI Documentation

- *Microsoft Telephony Application Programming Interface (TAPI) Programmer' s Reference* in the Microsoft Platform Software Development Kit, which documents Win32 TAPI applications.
- *Microsoft Telephony Service Provider Interface (TSPI) Reference* in the Microsoft Platform Service Provider Kit, which documents Win32 telephony service providers.

1. Introduction

1.1 What is the LDK TAPI Application and LDK TSP

The LDK TAPI is the application using Telephony functions based on LDK Keyphone System of LGE. And LDK TSP is the interface software or service provider that enables the application to access LDK Keyphone System.

Microsoft TAPI components included in Microsoft Windows use the routines in LDK TSP to access the system, and provide telephony functions to TAPI applications. Finally, the TAPI application can access LDK Keyphone System through the TAPI functions and LDK TSP routines.

The subsets of TAPI functions that LDK TSP supports include:

- Line Device Functions
- Phone Device Functions
- Device Specific Messages

For details of these functions, refer to Chapter 8 and associated documents described before.

Note: LDKSP is the filename of the LDK TSP.

Note: 1st party and 3rd party connection cannot be used at the same time.

1.2 Enabling the TAPI Applications to use LDK system

There are two modes enabling the TAPI applications to be serviced through LDK system. One is *the 3rd party mode* and the other is *the 1st party mode*.

In the 3rd party mode, referred to Figure 1.1, the TAPI application is on one of the client computers (may be Windows 95/98/2000/XP), which belongs to a Microsoft Windows domain serviced by the server computer (must be Windows NT Server 4.0 with Service Release 4 or later and Windows 2000/2003 Server). LDK TSP must be installed in the server computer, and the server computer must be connected to LDK system through the RS-232C port or LAN. The TAPI application can access LDK system through the Telephony RSP (Remote Service Provider) on the client computer through the network. Another TAPI application can access LDK system on another client computer.

In the 3rd party mode, the TAPI application on the client can access the line(s), which is(are) assigned to the client by the server computer.

In the 1st party mode, referred to Figure 1.2, the TAPI application is on the desktop computer (Windows 95/98/2000/XP), which is connected to the external CTI module or CTIU with the RS-232C port. The CTI Module/CTIU is connected to the keyset. The model names of keyset, which

can be used in the 1st party mode, are LDP or LKD Scout chip.

In the 1st party mode, the TAPI application on the desktop computer can access only one line, which is connected to the keyset. And, a few features of the lineDevSpecific function would not be supported.

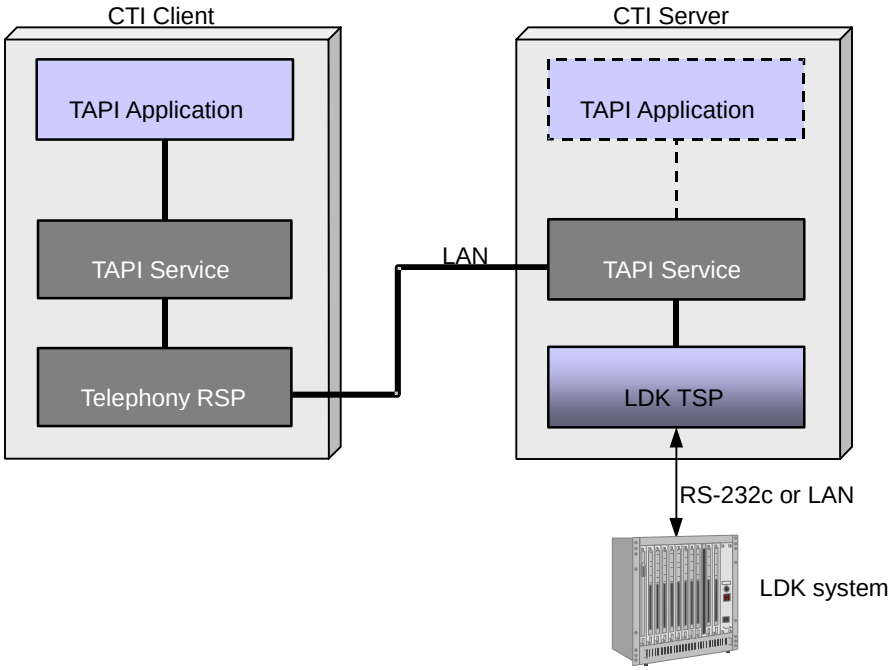


Figure 1.1 CTI Architecture of 3rd party mode

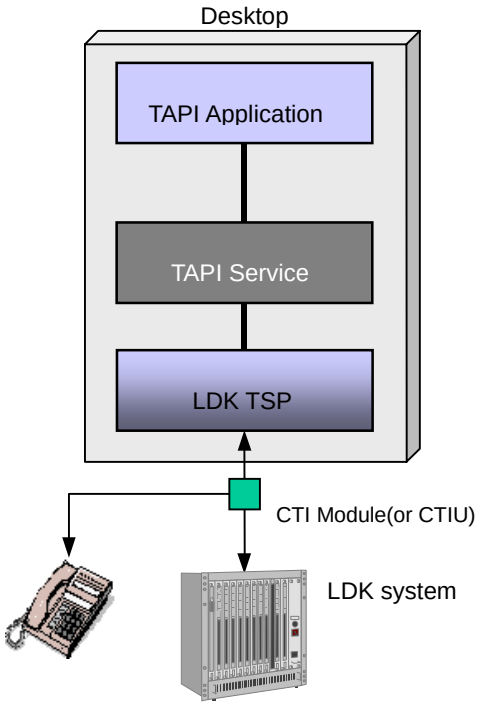


Figure 1.2 CTI Architecture of 1st party mode

2. Hardware/Software Requirements

2.1 Computer Requirements

The computer requirements are as follows:

- Server/Client (for 3rd party connection)

  Telephony Server: Windows NT 4.0(Service Pack 4 or later) / Windows 2000/2003 Server

  - CPU: Pentium III 500 MHz or later
  - RAM: 128 MB or later
  - Available Hard disk: 500 MB or later

  Client: Windows 95 or later (A Windows 95 must be upgraded to TAPI 2.1)

  - CPU: Pentium II 200 MHz or later
  - RAM: 32MB or later
  - Available Hard disk: 200 MB or later

2.2 LDK Key Telephone system Requirements

The LDK System requirements are as follows:

- System: LDK Key System (LDK-20/Nexer/50/100/300/600)
- To use LDK-50/100/300/600 1st party connection
  - CTI Module (Model:V70, LKD keyset with Scout chip)
  - CTIU enabled LDP keyset

- To use LDK-20/Nexer 1st party connection
  - Terminal Type

    DKT, SLT, WKT (GDC-345H or later, Middleware: 1.0I or later)
  - You need 1st party lock key or ez Phone lock key.

- To use 3rd party connection
  - Terminal Type

    DKT, SLT, WKT (GDC-345H or later, Middleware: 1.0I or later)
  - You need 3rd party lock key
  - For LDK-50/100, ASMU is needed.

2.3 LDK System Configurations

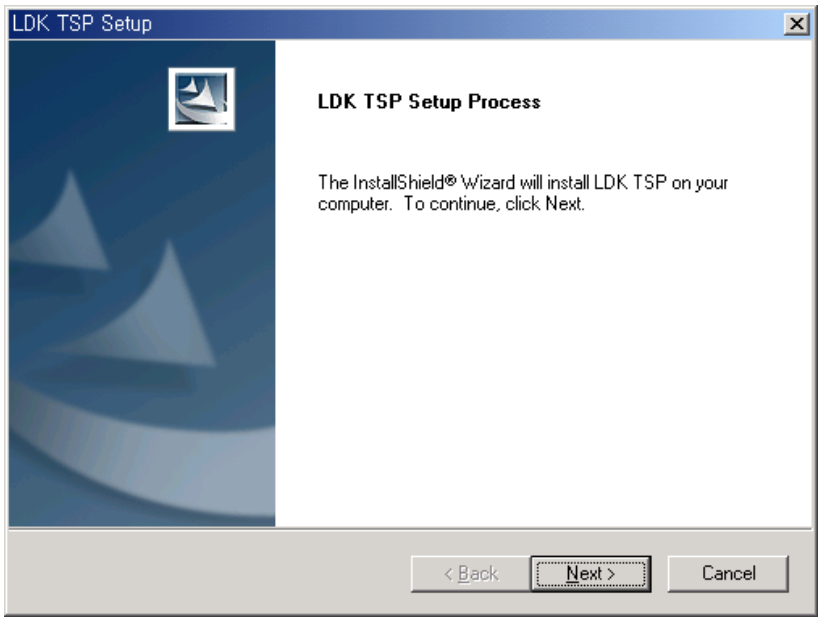
The LDK System Configurations are as follows:

- IP Setting ([TRANS/PGM] + 108)
- Connection Setting([TRANS/PGM] + 175 + Flex 11 + 02(COM2) or 12(TCP/IP))
- To use LDK-50/100/300/600 1st party connection

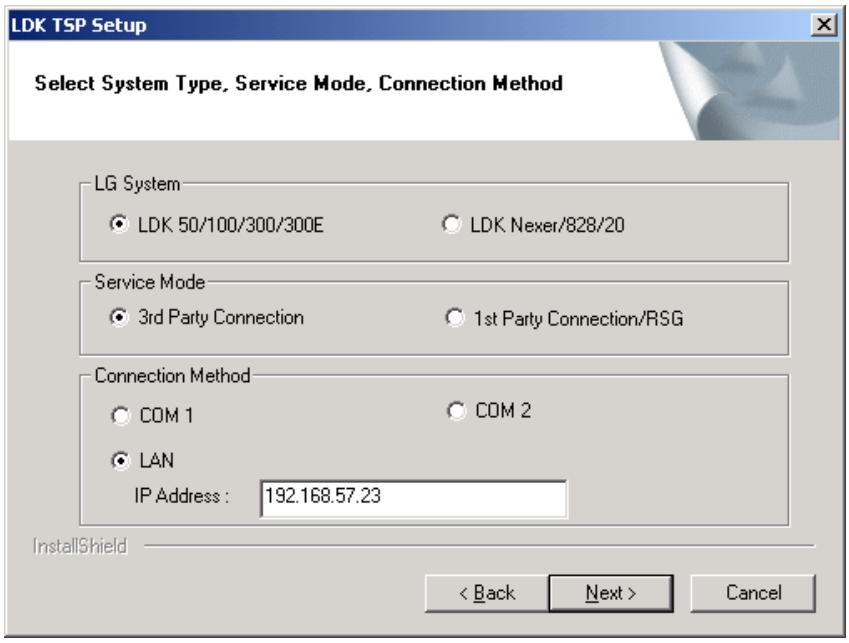
3. LDK TSP Installation & Configuration

3.1 Installation Procedure

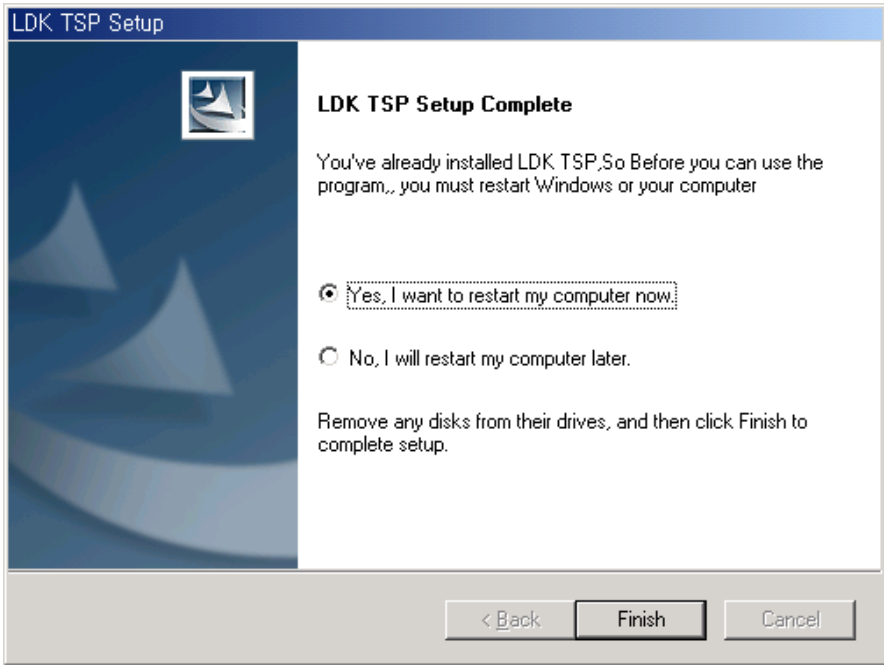
1. Run Setup.exe in LDK TSP install version.



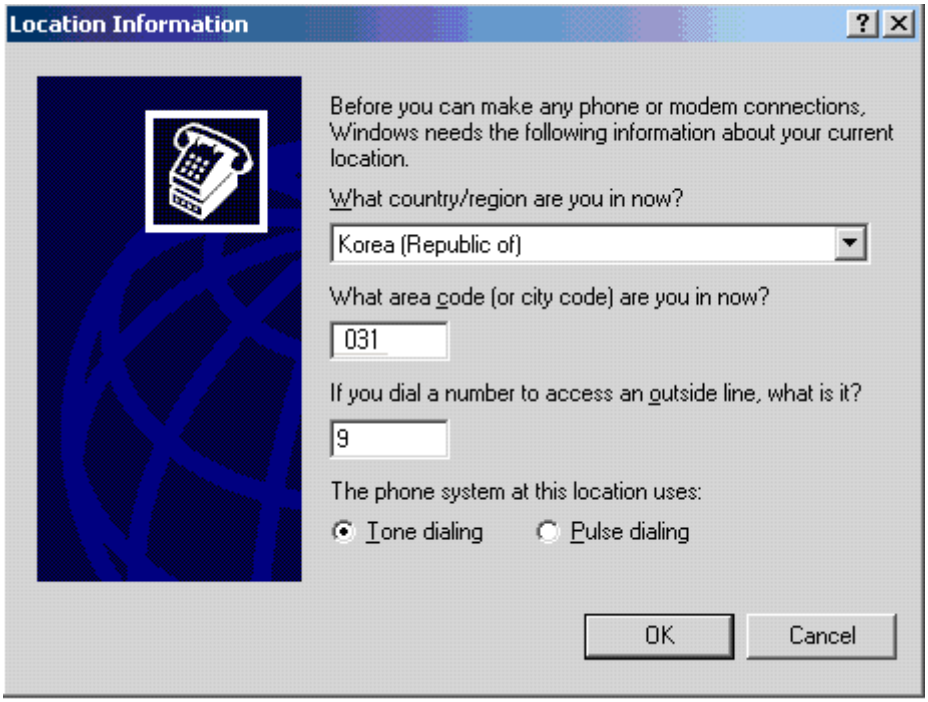
2. Click **[Next]** button.



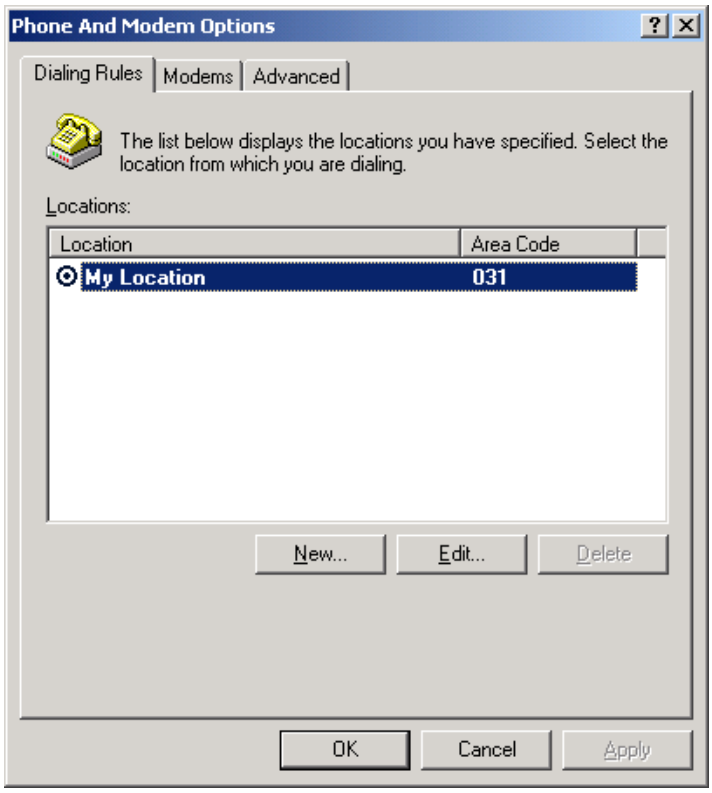
3. Select each item. When you select LAN connection, you should insert IP address of LDK system as above.



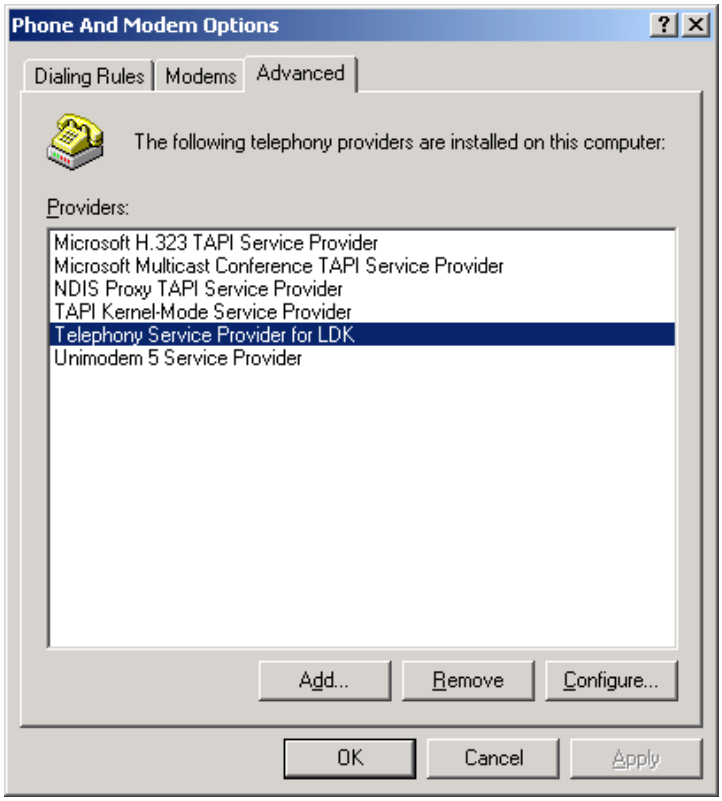
4. Restart your computer to connect between LDK TSP and LDK system.
5. You can change the LDK TSP configurations as following
6. Open **[Phone and Modem Options]** in Control Panel.



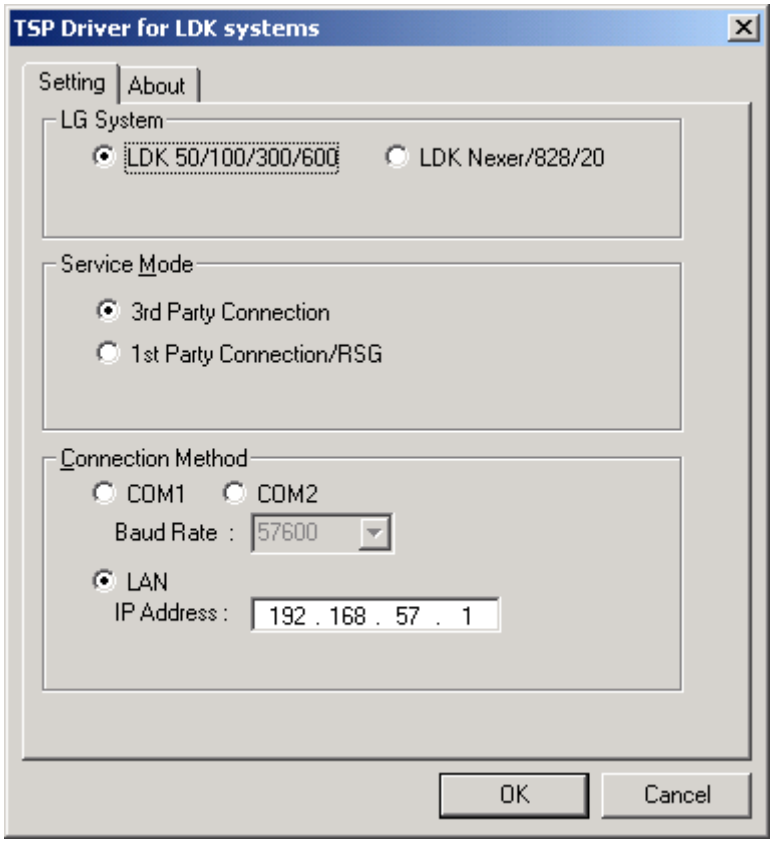
- Select country/region, insert area code and outside dial number.



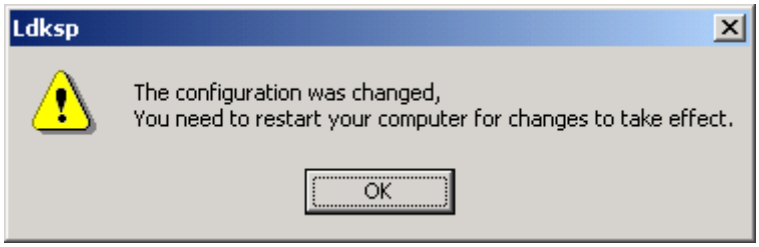
7. On the **[Advanced]** tab under Providers, click **[Telephony Service Provider for LDK]**.



8. Click **[Configure…]**.



9. You can change LDK TSP configurations in this dialog.



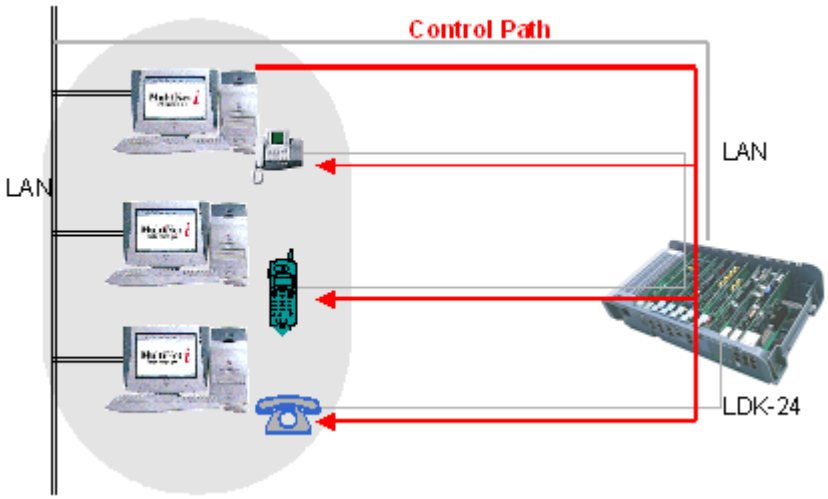
10. If the configurations were changed, you should restart your computer to take effect.

4. 1st party connection

A computer is communicating with LDK system directly. You should install LDK TSP on your local computer.

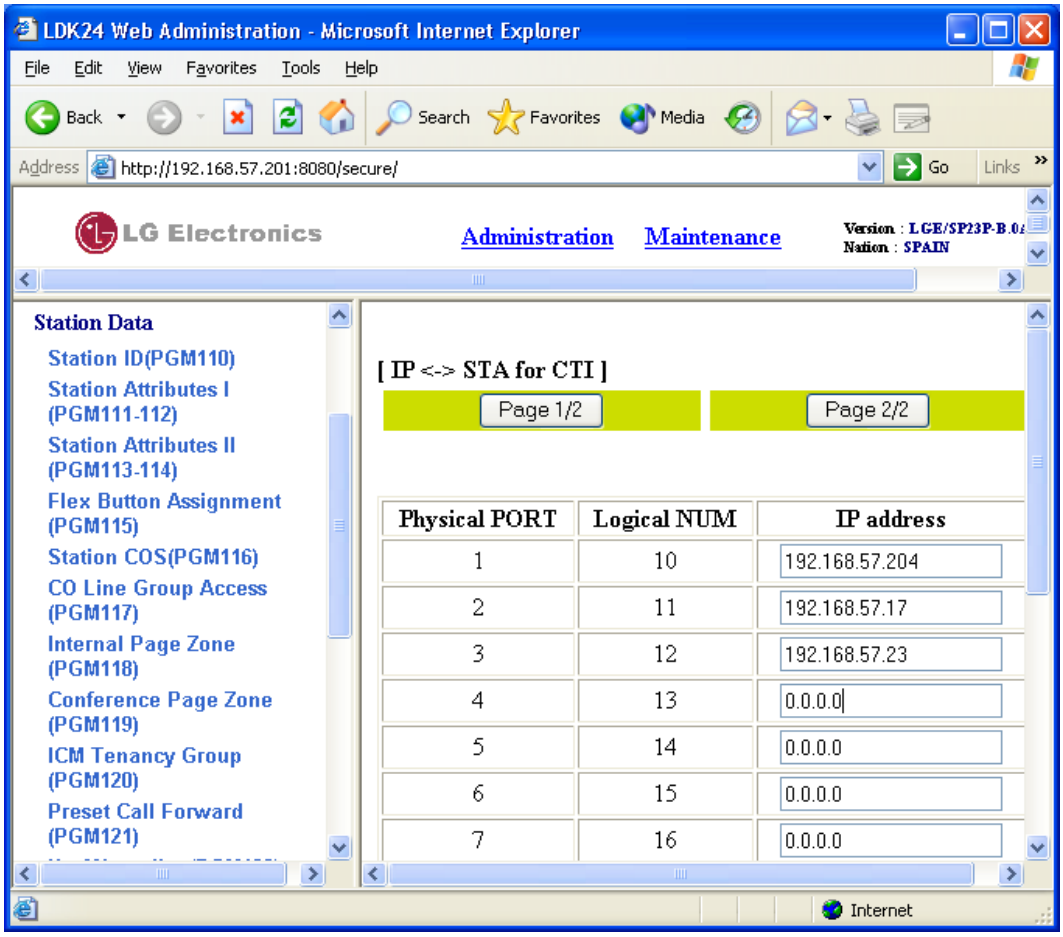
4.1 LDK-20/Nexer

4.1.1. System configurations



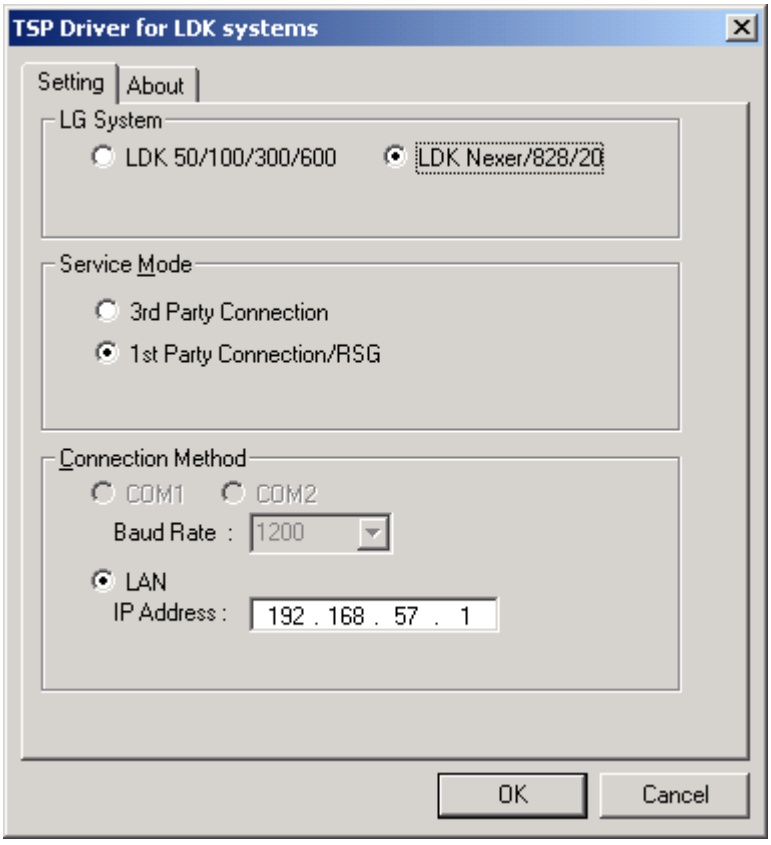
4.1.2. To assign a user to a line

1. In PGM 125, type a client IP address in Edit box, click **[Save]** button.



4.1.3. To configure a LDK TSP

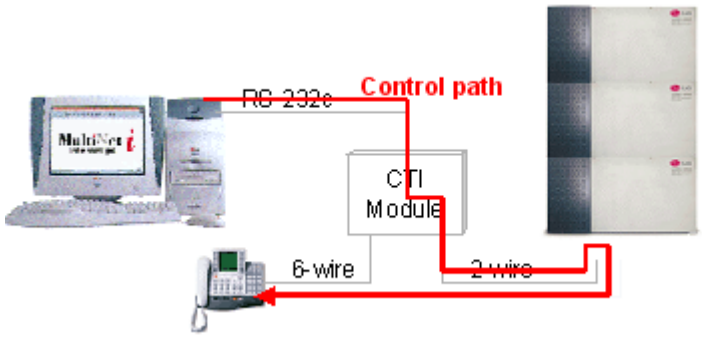
1. Verify LDK TSP configurations.



- Select LDK Nexer/828/20 in LG System.
- Select RSG/1st party Connection in Service Mode.
- Select LAN in Connection Method, and type IP address of LDK system.

4.2 LDK-50/100/300/600

4.2.1. System configurations



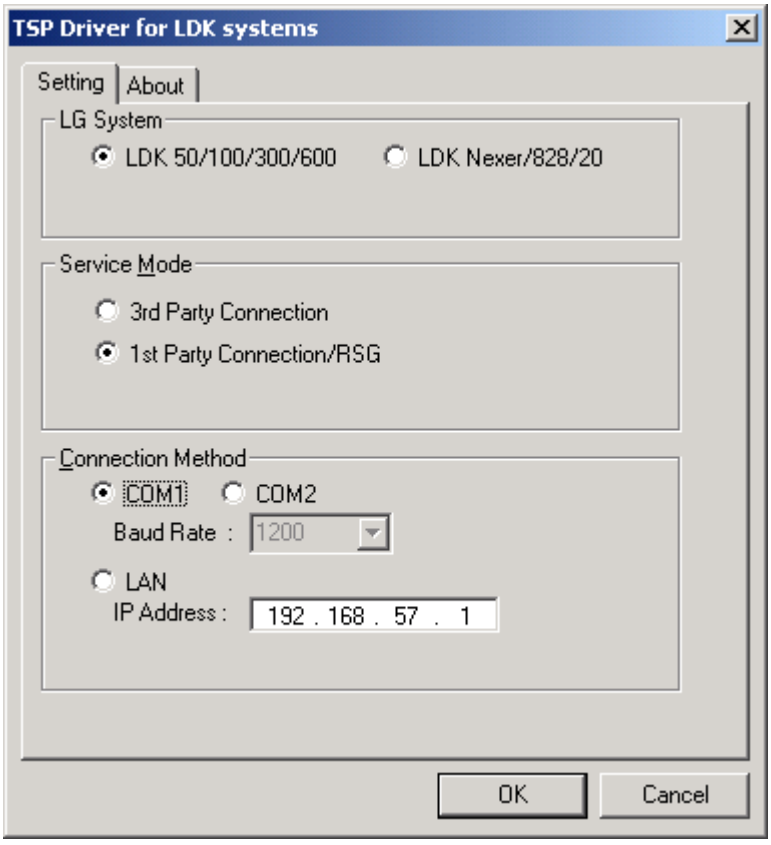
[Figure, How to use CTI Module]

[Figure, How to use CTIU enabled keyset]

- In case of using CTI module, CTI module (Model: V70) is connected to LKD with Scout chip.
- CTIU enabled LDP keyset is available.

4.2.2. To configure a LDK TSP

1. Verify LDK TSP configurations

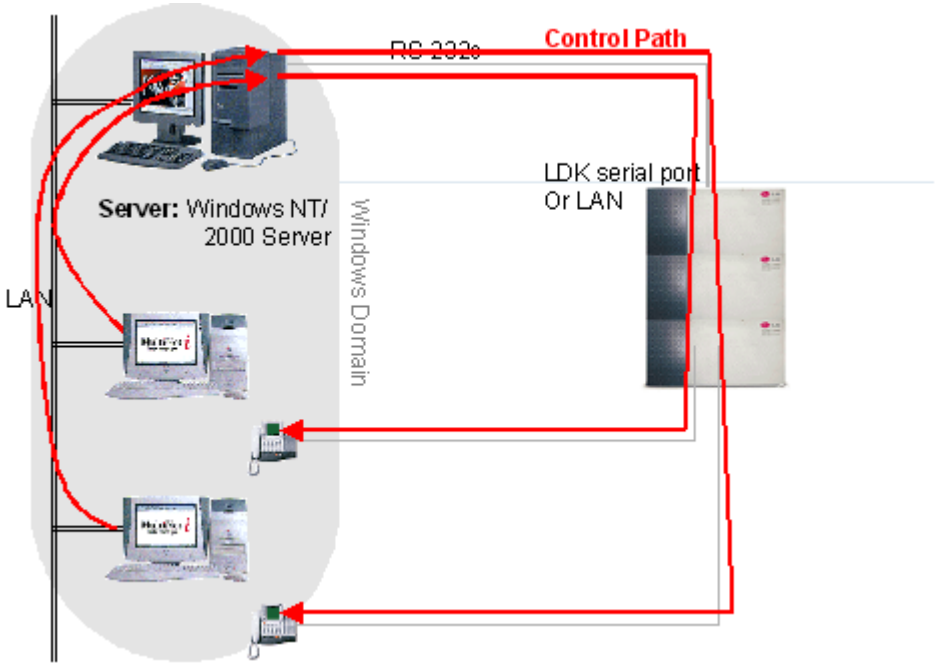


- Select LDK-50/100/300/600 in LG System
- Select 1st party Connection/RSG in Service Mode
- Select COM1 or COM2, In case of RSG, LAN connection is available.

5. 3rd party connection

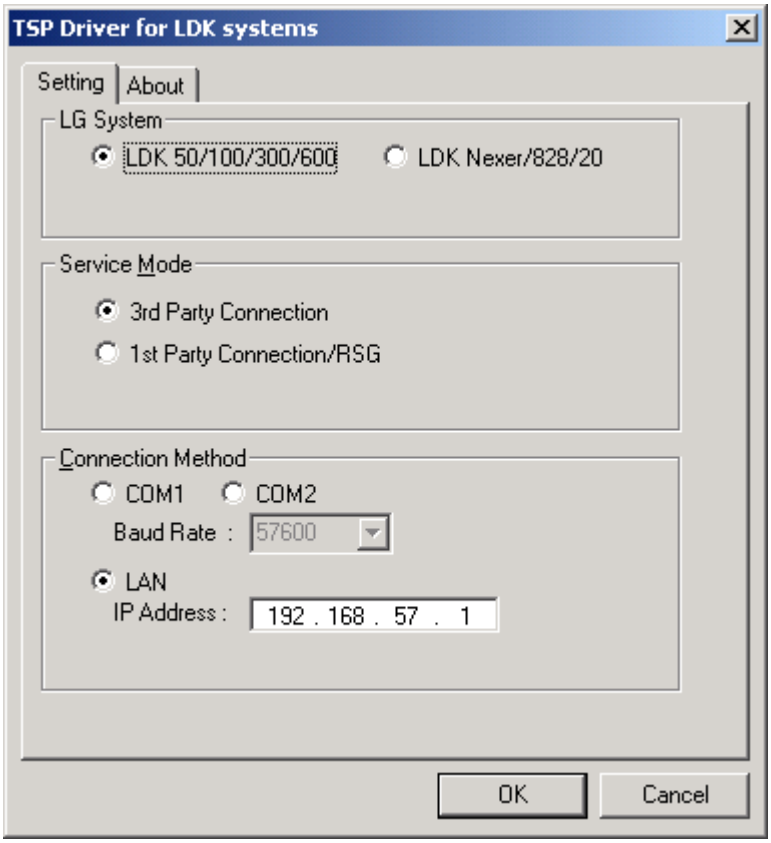
For 3rd party connection, you need a Server computer (Windows 2000/2003 Server or Windows NT 4.0). You should install LDK TSP on Server computer. In client PC, TAPI Application is communicating with its server through RSP (Remote Service Provider).

5.1 System Configurations



5.2 To configure a LDK TSP

1. Verify LDK TSP configurations



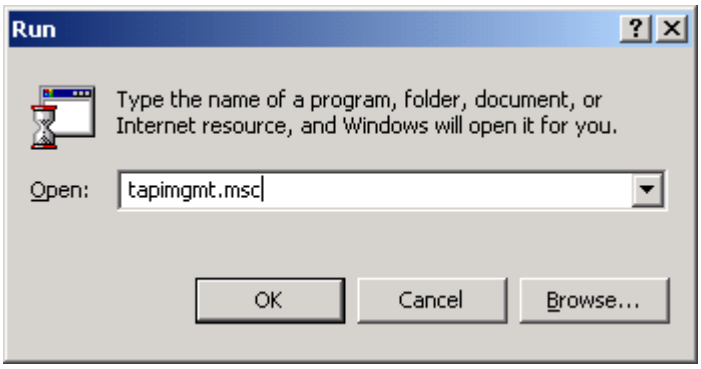
- Select System.
- Select 3rd party connection
- You can use anyone of connection Method.

6. Telephony Server Configurations

6.1 To enable or disable a telephony server

Windows 2000/2003 Server

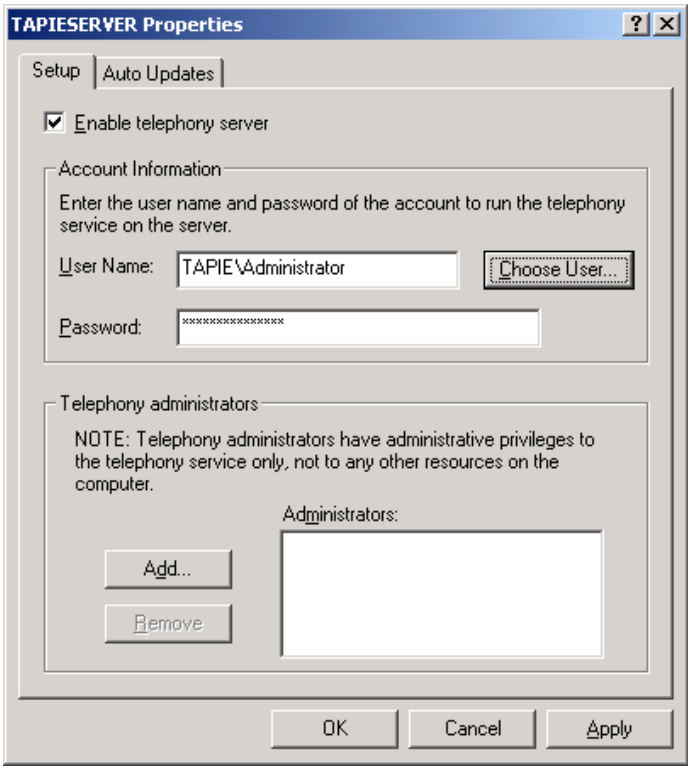
1) Open **Telephony**



- To open **Telephony** on Windows 2000/2003 Server, click **Start**, click **Run**, and then type **tapimgmt.msc**.

2) In the console tree, click the server you want to manage.

3) In the **Action** menu, click **Properties**.



4) On the **Setup** tab, select the **Enable telephony server** check box to enable the server or clear the check box to disable the server.

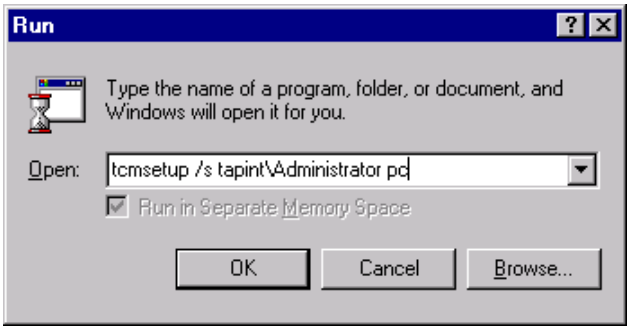
5) When you enable the server, specify a logon account under **Account Information**. It must be a member of the Administrators group on the server, and also be in the same domain as the server or in a domain that has a two-way trust relationship with the domain containing the server.

This procedure changes whether clients can use the telephony devices on the server. Changes take effect the next time the system attempts to start the TAPI service.

If the **Enable telephony server** check box is unavailable, you need to start the Telephony service first. To manage the Telephony service, you must be a telephony administrator on the server or logged on as an administrator of the server.

Windows NT 4.0

1) Click **Start**, Click **Run**.

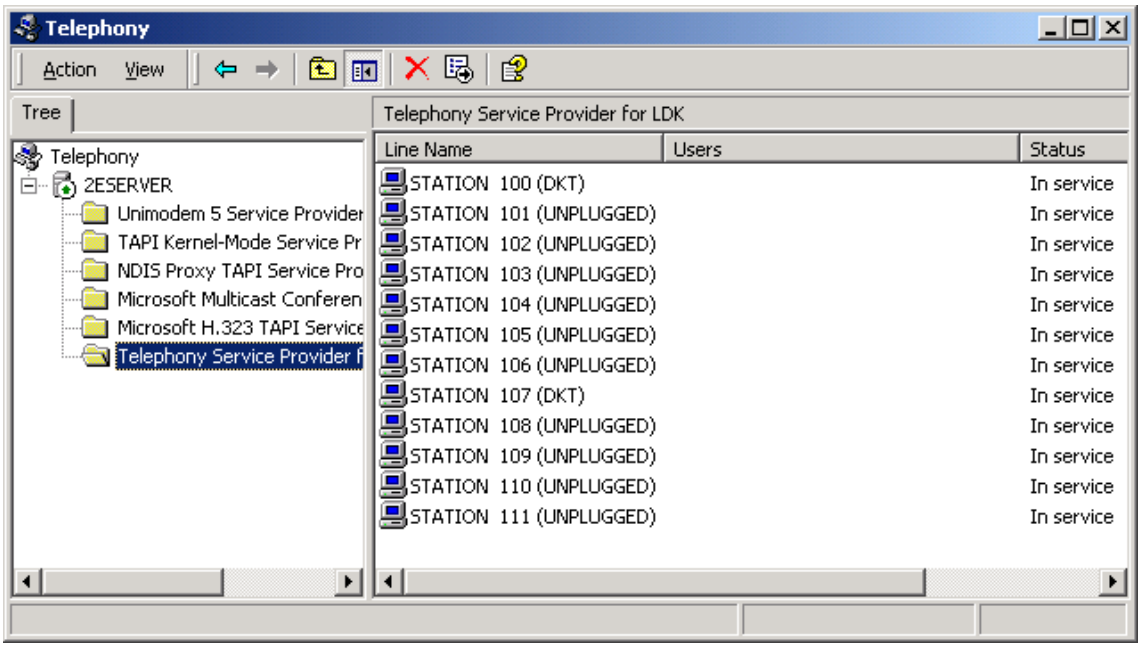


- tcmsetup /s domain\username [password]

6.2 To assign a telephony user to a line or phone

Windows 2000/2003 Server

1) Open **Telephony**
2) In the console tree, click **Telephony Service Provider for LDK**.



3) In the details pane, under **Line Name** or **Phone Name**, click the line or phone.
4) In the **Action** menu, click **Edit Users**, and then click **Add**.
5) In **Look in**, click the domain containing the user.
6) In the list, select the user, and then click **Add**.

This procedure permits TAPI client programs run by the user to use the line or phone on the server. A user added for a phone or line must be in the same domain as the TAPI server or in a domain that has a two-way trust relationship with the domain containing the server. Before a user can use the telephony line or phone, the TAPI server must be specified on the client computer.

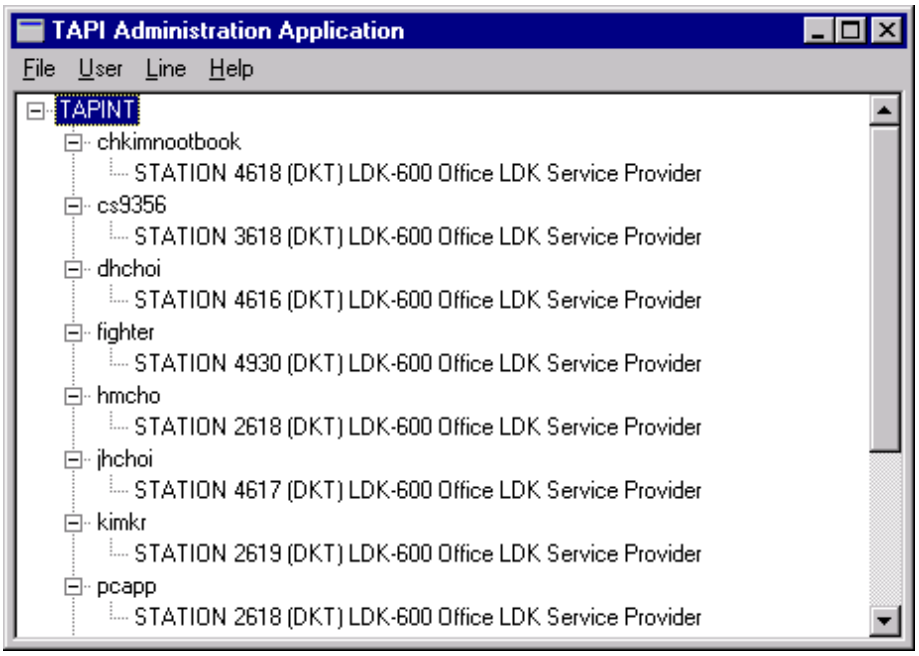
Windows NT 4.0

1) Open **TAPI Administration Application**.

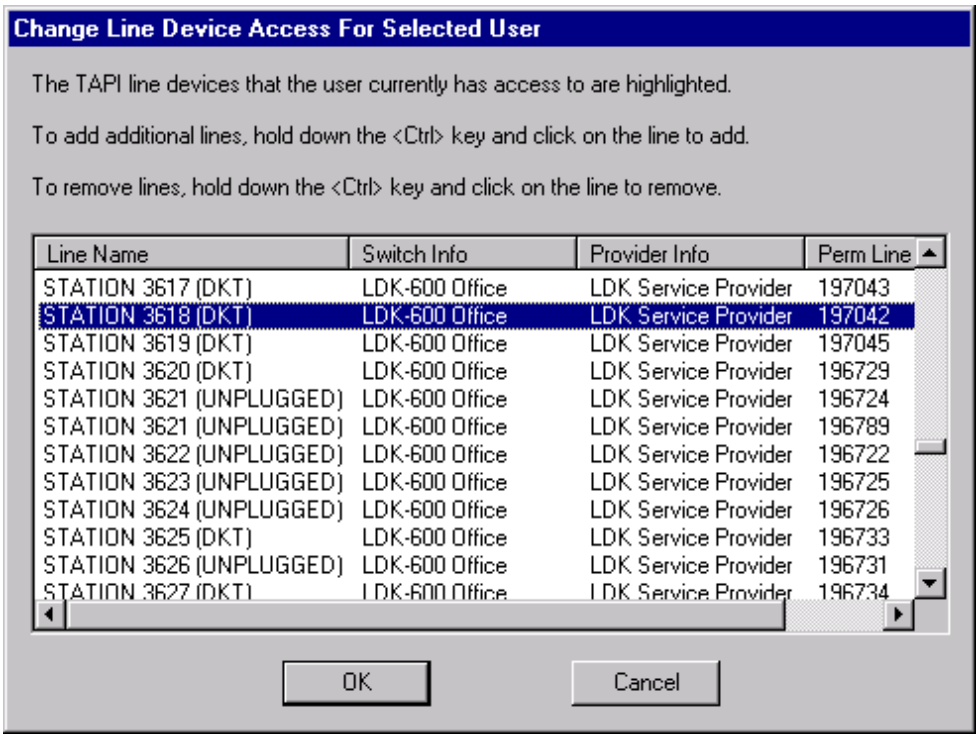
- To open **TAPI Administration Application** on Windows NT 4.0, click **Start,** click **Run**, and then type **tcmapp**.

2) In the **User** menu, click **Add a User to this Telephony Server**.

3) In the list, select the user.



4) Click user, in the **User** menu, click **Change Line Device Access for Selected User**



5) Select the line, click **OK**.

6.3 To remove users from telephony lines or phones

Windows 2000/2003 Server

1) Open Telephony.
2) In the console tree, click Telephony Service Provider for LDK.
3) In the details pane, under **Line Name** or **Phone Name**, click the line or phone.
4) In the **Action** menu, click **Edit Users**.
5) In **Assigned Users**, click a user, and then click **Remove**.

After you perform this procedure, TAPI client programs run by the user will be unable to use the line or phone on the TAPI server.

Windows NT 4.0

1) Open TAPI Administration Application
2) Select the line, click **Remove Selected User**.

7. Client Computer Configurations

7.1 To specify telephony servers on a client computer

Windows 2000/XP

1) Open a **Command Prompt** window.
2) Type **Tcmsetup /c** server1 server2….

To open a Command Prompt window, click **Start**, point to **Programs**, point to **Accessories**, and then click **Command Prompt**.

The TAPI client is installed with the Microsoft Windows 2000 operating system, but you need to perform this procedure to specify servers for the client. The client must be in the same domain as the server or in a fully trusted domain.

You must log on to the client as an administrator to perform this command. If you are logged on to a computer in the Users or Power Users group, you can use the **runas** command to run **tcmsetup** as an administrator, for example:

- runas /user:mydomain\myname " tcmsetup /c servername"

You may have to restart the Telephony service for this change to take effect.

When you use Windows Server 2003, the following service pack should be installed on a client computer.

- Windows 2000 Professional: Service Pack 5 or later.
- Windows XP Home: Service Pack 2 or later.
- Windows XP Professional: Service Pack 2 or later.

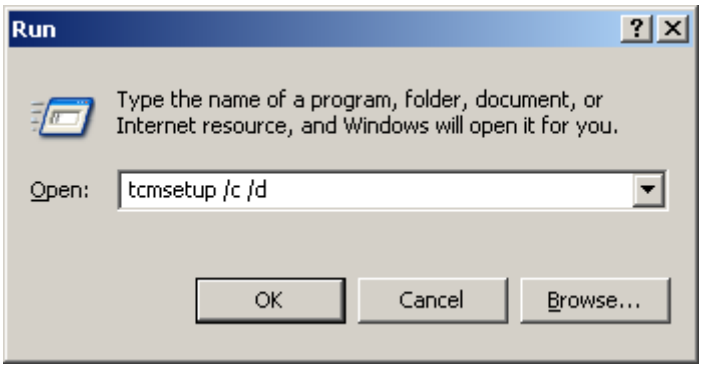
Windows 95, 98

1) On a Windows 95 or Windows 98 TAPI client computer, click **Start**, point to **Settings**, and then click **Control Panel**.
2) Double-click **Network**.
3) (Optional) If **Client for Microsoft Networks** is not in the list of installed network components, click **Add**, and follow the instructions to add it.
4) Click **File and Print Sharing**, conform that both check boxes are selected, and then click **OK**.
5) Click the **Access Control** tab, click **User-level access control**, and then click **OK**.
6) Open a **Command Prompt** window, type **tcmsetup /c** servername.

After you complete this procedure, you need to specify a TAPI server on the client. To make the telephony resources on the server available to the client users, the server administrator needs to assign the users to telephones or lines on the server. Users must log onto a valid domain account. A Windows 95 TAPI client must be upgraded to TAPI2.1 to use the TAPI services provided by a Windows 2000 TAPI server.

7.2 Client disable from telephony servers

1) Open a **Command Prompt** window.



- tcmsetup /c /d

Client user cannot see the lines or phones.

8. Support For TAPI Functions

8.1 Line Device Functions

The following Sections describe TAPI Line Device functions supported by LDK TSP.

For more information about TAPI functions, refer to the *Microsoft Windows Telephony Application Programmer's Guide*.

TAPI Function	LDK TSP Support
lineAddProvider	Supported (TAPI itself).
lineAddToConference	Supported.
lineAnswer	Supported.
lineBlindTransfer	Supported.
lineClose	Supported.
lineCompleteCall	Supported.
lineCompleteTransfer	Supported.
lineConfigProvider	Supported (TAPI itself).
lineDeallocateCall	Supported (TAPI itself).
lineDevSpecific	Supported.
lineDial	Supported.
lineDrop	Supported.
lineForward	Supported.
lineGenerateDigits	Supported.
lineGetAddressCaps	Supported.
lineGetAddressID	Supported.
lineGetAddressStatus	Supported.
lineGetCallInfo	Supported.
lineGetCallStatus	Supported.
lineGetConfRelatedCalls	Supported.
lineGetCountry	Supported (TAPI itself).
lineGetDevCaps	Supported.
lineGetID	Supported.
lineGetLineDevStatus	Supported.
lineGetMessage	Supported (TAPI itself).
lineGetNewCalls	Supported.

lineGetProviderList	Supported (TAPI itself).
lineGetStatusMessages	Supported.
lineHold	Supported.
lineInitializeEx	Supported.
lineMakeCall	Supported.
lineNegotiateAPIVersion	Supported.
lineNegotiateExtVersion	Supported.
lineOpen	Supported.
TAPI Function	**LDK TSP Support**
linePark	Supported.
linePickup	Supported.
linePrepareAddToConference	Supported.
lineRedirect	Supported.
lineReleaseUserUserInfo	Supported.
lineRemoveFromConference	Supported.
lienRemoveProvider	Supported (TAPI itself).
lineSendUserUserInfo	Supported.
lineSetCallData	Supported.
lineSetCallPrivilege	Supported.
lineSetStatusMessages	Supported.
lineSetupConference	Supported.
lineSetupTransfer	Supported.
lineShutdown	Supported.
lineSwapHold	Supported.
lineTranslateAddress	Supported (TAPI itself).
lineTranslateDialog	Supported (TAPI itself).
lineUncompleteCall	Supported.
lineUnhold	Supported.
lineUnpark	Supported.

The followings are more detail descriptions of LDK-supporting TAPI functions. It was referred to MSDN Library.

lineAnswer

The **lineAnswer** function answers the specified offering call. The function is completed asynchronously.

```
LONG lineAnswer(
   HCALL hCall,
   LPCSTR lpsUserUserInfo,
   DWORD dwSize
);
```

- LDK TSP supports the user-user information in the lpsUserUserInfo.
- SLT cannot use this function.

lineCompleteCall

The **lineCompleteCall** function specifies how a call that could not be connected normally should be completed instead. The network or switch may not be able to complete a call because network resources are busy or the remote station is busy or doesn't answer. The application can request that the call be completed in one of a number of ways. The function is completed asynchronously.

```
LONG lineCompleteCall(
   HCALL hCall,
   LPDWORD lpdwCompletionID,
   DWORD dwCompletionMode,
   DWORD dwMessageID
);
```

- **LINECALLCOMPLMODE_CALLBACK:** A user can use this completion mode when the called user does not answer or the station status of the called user is DND(Do Not Disturb)
- **LINECALLCOMPLMODE_CAMPON:** A user can use this completion mode when the station status of t he called user is busy.
- **LINECALLCOMPLMODE_INTRUDE:** When the called user does not answer the call, the Attendant user can use this completion mode.
- **LINECALLCOMPLMODE_MESSAGE**: When the called user does not answer the call, the caller user leaves a voice message.

lineDevSpecific

The **lineDevSpecific** function enables service providers to provide access to features not offered by other TAPI functions. The meanings of the extensions are device specific, and taking advantage of these extensions requires the application to be fully aware of them. The function is completed asynchronously.

```
LONG lineDevSpecific(
   HLINE hLine,
   DWORD dwAddressID,
   HCALL hCall,
   LPVOID lpParams,
   DWORD dwSize
);
```

- dwAddressID must be zero.

LDK TSP supports the following device-specific functions. Some features are supported only in the 3rd party mode:

- **Paging**

  " PAGE/xx…x" in the lpParams field where xx…x is:

  1. INT_PAGE_Z01 ~ INT_PAGE_Z35: Announce your voice to the internal zone (01~35).
  2. INT_ALL_CALL: Announce your voice to all internal zones.
  3. EXT_PAGE_Z01 ~ EXT_PAGE_Z03: Announce your voice to the external zone (01~03).
  4. EXT_ALL_CALL: Announce your voice to all external zones.
  5. ALL_CALL: Announce your voice to all external and internal zones.
  6. MEET_ME: Answer and connect the announcing call.

  This function can be used when the call (hCall) is in the following status.

  - in an idle status
  - in an dial status
  - in the status of hearing a dial tone, an error tone, or a busy tone

  If a wrong lpParams is transferred, or the line device does not have a page access right, or anyone is using a page function, or there is no member in the transferred page zone, the lResult actual parameter of the corresponding ASYNC_COMPLETION is LINEERR_OPERATIONFAILED. The function is completed successfully

  Ex) If you want to announce your voice to all internal zones, include the following codes to your application source code. Most device-specific functions use the similar method to the following source code.

  ```
  char lpParam[]=" PAGE/INT_ALL_CALL" ;
  lineDevSpecific( hLine, dwAddressID, hCall, (LPVOID)lpParam, sizeof(lpParam) );
  ```

- **DVU (Digitized Voice Unit)**

  " DVU/xx…x" in the lpParams field where xx…x is:

  1. DATE; voice announcement of the date and the time.

2. STANO; voice announcement of the number of the station
3. OUTMSG_REC; record customer message
4. OUTMSG_DEL; delete customer message
5. OUTMSG_PLAY; play customer message
6. STA_STATUS; voice announcement of the following information of the station
   - Station Number
   - ICM Signaling Mode
   - Number of Message Waiting
   - Wake-up Time
   - DND state
   - Forwarded to other station
   - Forwarded to Speed Bin
   - Station COS X
7. PAGEMSG_REC; record page announcement
8. PAGEMSG_DEL; delete page announcement
9. PAGEMSG_PLAY; play page announcement
10. MSGWAIT_PLAY; play the message queued
11. MSGWAIT_DEL; delete the current message
12. MSGWAIT_PLAY_NEXT; play next message queued
13. MSGWIAT_REW: rewind the message due to admin setting.
14. MSGWAIT_ADD: add voice message.
15. MSGWAIT_CALLBACK: With CLI, make call to an external.
16. FORWARD_UNCOND: set incoming call to forward to DVU unconditionally.
17. FORWARD_BUSY: set incoming call to forward to DVU when It is busy.
18. FORWARD_NOANSW: set incoming call to forward to DVU when it is no answer
19. FORWARD_BUSYNA: set incoming call to forward to DVU when it is busy and no answer.

This function with lpParams not related to Message-waiting can be used in an idle status or in the status of hearing a dial tone. This function with lpParams of MSGWAIT_PLAY can be used in an idle status or the status of hearing a dial tone. If the application uses this when hearing a DVU message, the message will be replayed.
If a wrong lpParams is transferred, or the line device does not have a DVU access right, or DVU board is not installed, or there is not available channel, or memory is insufficient, the

lResult actual parameter of the corresponding ASYNC_COMPLETION is LINEERR_OPERATIONFAILED.

- **Get SMDR (Station Message Detail Recording) Data**

  " SMDR/x" in the lpParams field where x is:

  1. B (Begin)
  2. E (End)

  This function enables or disables the application to receive the SMDR data. If the application uses this function with lpParams of " SMDR/B" , the LDK TSP will send the application the device-specific message of SMDR whenever the SMDR data occurs from LDK system, and then, you can get the SMDR data using **lineGetAddressStatus** function. If the application doesn' t want to receive any SMDR data from LDK, use this function with lpParams of " SMDR/E" .

  ***Note**: 3rd party mode only*

- **Get Changed Station Status**

  " STA_STATUS/x" in the lpParams field where x is:

  1. B (Begin)
  2. E (End)

  This function enables or disables the application to receive the changed station status data. If the application uses this function with lpParams of " STA_STATUS/B" , the LDK TSP will send the application the device-specific message whenever any stations' status is changed. The message will have the physical number of the station and the changed status data in its parameters. Refer to LINE_DEVSPECIFIC message of the status. If the application doesn' t want to receive any station status data from LDK TSP, use this function with lpParams of " STA_STATUS/E" .

  The application can monitor the status changing of any station connected to the LDK system, even if the line of a station doesn' t be open by the application.

  ***Note**: LDK-50/100/300/600 1st party does not support*

- **Get Changed CO Line Status**

  " COL_STATUS/x" in the lpParams field where x is:

  1. B (Begin)
  2. E (End)

  This function enables or disables the application to receive the CO line status data. If the application uses this function with lpParams of " COL_STATUS/B" , the LDK TSP will send

the application the device-specific message whenever the CO line status is changed. The message will have the physical number of the CO line and the changed status data in its parameters. Refer to LINE_DEVSPECIFIC message of the status. If the application doesn't want to receive any CO line status data from LDK TSP, use this function with lpParams of "COL_STATUS/E".

The application can monitor the status changing of any CO line connected to the LDK system.

***Note**: LDK-50/100/300/600 1st party does not support*

- **Get All Station Status**

  "ALL_STA_STATUS" in the lpParams field

  This function can tell the application the status of all stations. The status is described in 0 Device Specific Data Format. After the function is completed asynchronously, the LINE_DEVSPECIFIC message is sent to the application. Then, the application can get the status data using **lineGetAddressStatus** function. Refer to LINE_DEVSPECIFIC message and **lineGetAddressStatus** function.

  The application can get the status of any station connected to the LDK system, even if the line of a station doesn't be open by the application.

  ***Note**: LDK-50/100/300/600 1st party does not support*

- **Get All CO Line Status**

  "ALL_COL_STATUS" in the lpParams field

  This function can tell the application the status of all CO lines. The status is described in 0 Device Specific Data Format. After the function is completed asynchronously, the LINE_DEVSPECIFIC message is sent to the application. Then, the application can get the status data using **lineGetAddressStatus** function. Refer to LINE_DEVSPECIFIC message and **lineGetAddressStatus** function.

  The application can get the status of any CO line connected to the LDK system.

  ***Note**: LDK-50/100/300/600 1st party does not support*

- **Get System Time**

  "GETTIME" in the lpParams field

  This function can tell the application current LDK TSP server time. After the function is completed asynchronously, the LINE_DEVSPECIFIC message is sent to the callback function. Refer to LINE_DEVSPECIFIC.

  ***Note**: 3rd party mode only*

- **Set CO Line Hold Type**

  " CO_HOLD/xxx" in the lpParams field where xxx is:

  1. EXC(Exclusive Hold)
  2. SYS(System Hold)

  This function sets the holding type of CO line. If an application holds a CO line using **lineHold**, the CO line will be held with the type that the application sets using this function. For example, after an application uses this function with " CO_HOLD/SYS" in the lpParams field, the CO line held with the function Ошибка! Источник ссылки не найден. will be on system holding. The default type of holding is Exclusive Hold.

- **Get Station' s Logical Number**

  " STA_NUM/xxx" in the lpParams field where " xxx" is the physical number of the station (eg. 001, 010, 100)

  This function asks LDK TSP to return the logical number (station number) of the station. After asynchronous successful response of this function, the application will receive the LINE_DEVSPECIFIC message, which contains the physical number and the logical number of the station in its parameters. Refer to LINE_DEVSPECIFIC.

  ***Note**: LDK-50/100/300/600 1st party does not support*

- **Get All Stations' Logical Numbers**

  " ALL_STA_NUM" in the lpParams field

  This function asks LDK TSP to return the logical numbers (station number) of all stations. After asynchronous successful response of this function, the application will receive the LINE_DEVSPECIFIC message. Then, the application can get the logical numbers of all stations by using **lineGetAddressStatus** function. Refer to LINE_DEVSPECIFIC message and **lineGetAddressStatus** function.

  ***Note**: LDK-50/100/300/600 1st party does not support*

- **Get Station Information**

  " STA_INFO/xxx" in the lpParams field where " xxx" is the physical number of the station (eg. 001, 010, 100)

  This function asks LDK TSP to prepare to provide the information of the station, which has the physical number of " xxx." After asynchronous successful response of this function, the application will receive the LINE_DEVSPECIFIC message. Then, the application can get the information of the station by using **lineGetAddressStatus** function. Refer to LINE_DEVSPECIFIC message and **lineGetAddressStatus** function.

  ***Note**: LDK-50/100/300/600 1st party does not support*

- **Get CO Line Information**

  " COL_INFO/xxx" in the lpParams field where " xxx" is the physical number of the CO line (eg. 001, 010, 100)

  This function asks LDK TSP to prepare to provide the information of the CO line, which has the physical number of " xxx." After asynchronous successful response of this function, the application will receive the LINE_DEVSPECIFIC message. Then, the application can get the information of the station by using **lineGetAddressStatus** function. Refer to LINE_DEVSPECIFIC message and **lineGetAddressStatus** function.

  ***Note**: LDK-50/100/300/600 1st party does not support*

- **Get All Park Number**

  " ALL_PARK_NUM" in the lpParams field.

  This function asks LDK TSP to prepare to provide the information of all park number. After asynchronous successful response of this function, the application will receive the LINE_DEVSPECIFIC message. Then, the application can get the information of the station by using **lineGetAddressStatus** function. Refer to LINE_DEVSPECIFIC message and **lineGetAddressStatus** function.

  ***Note**: 3rd party mode only*

- **Dummy Station Log on**

  " LOGON/xxxx/yyyyy" in the lpParams field where xxxx, yyyyy are:.

  xxxx: Station Number (3 or 4 digits)

  yyyyy: Password (5 digits)

- **Dummy Station Log off**

  " LOGOFF/xx…xx" in the lpParams field where xx…xx is:.

  - NOCHANGE
  - VMIB
  - VMGROUP/yyyy, yyyy is a VM group number
  - VMIB
  - MOBILE
  - STATION/xxxx, xxxx is a station number. Forward to Station.

- **UCD DND Log on/off in UCD group**

  " UCD_DND_ON/xxxx" in the lpParams: Activate UCD DND.

  " UCD_DND_OFF/xxxx" in the lpParams: Deactivate UCD DND.

- xxxx is a UCD group number.

" S_UCD_DND_ON/xxxx/yyyy" in the lpParams: Activate UCD DND by Supervisor.
" S_UCD_DND_OFF/xxxx/yyyy" in the lpParams: Activate UCD DND by Supervisor.
- xxxx is a UCD group number.
- yyyy is a station number (3 or 4 digits)

- **Make/delete Conference Room**
  " CONF_ROOM/E/xxx/yyyyy" in the lpParams: Activate Conference Room.
  " CONF_ROOM/E/xxx/yyyyy" in the lpParams: Deactivate Conference Room.
  - xxxx is a Conference Room number.
  - yyyyy is a password.

lineDial

The **lineDial** function dials the specified dialable number on the specified call. The function is completed asynchronously.

```
LONG lineDial(
  HCALL hCall,
  LPCSTR lpszDestAddress,
  DWORD dwCountryCode
);
```

- **Account Code**
  " AC/xx…x" in the lpszDestAddress field where xx…x are account code digits which are within 12 digits.
  These digits will be included in the SMDR data record. These can be used to allocate cost or time for the call.
  This function can be used when the call (hCall) is in the connected status.

lineForward

The **lineForward** function forwards calls destined for the specified address on the specified line, according to the specified forwarding instructions. When an originating address (dwAddressID) is forwarded, the specified incoming calls for that address are deflected to the other number by the switch. This function provides a combination of forward and do-not-disturb features. This function can also cancel forwarding currently in effect. The function is completed asynchronously.

```
LONG lineForward(
   HLINE hLine,
   DWORD bAllAddresses,
   DWORD dwAddressID,
   LPLINEFORWARDLIST const lpForwardList,
   DWORD dwNumRingsNoAnswer,
   LPHCALL lphConsultCall,
   LPLINECALLPARAMS const lpCallParams
);
```

- dwAddressID must be zero.
- LDK TSP supports the following functions:

- **DND (Do Not Disturb)**
  lpForwardList→ForwardList[0].dwDestAddressSize to NULL.
- **Forward all calls unconditionally.**
  lpForwardList→ForwardLIst[0].dwForwardMode to LINEFORWARDMODE_UNCOND.
- **Forward any calls on busy.**
  lpForwardList→ForwardLIst[0].dwForwardMode to LINEFORWARDMODE_BUSY.
- **Forward any calls on no answer.**
  lpForwardList→ForwardLIst[0].dwForwardMode to LINEFORWARDMODE_NOANSW.
- **Forward any calls on busy/no answer.**
  lpForwardList→ForwardLIst[0].dwForwardMode to LINEFORWARDMODE_BUSYNA.

lineGetAddressStatus

The **lineGetAddressStatus** function allows an application to query the specified address for its current status.

```
LONG lineGetAddressStatus(
   HLINE hLine,
   DWORD dwAddressID,
   LPLINEADDRESSSTATUS lpAddressStatus
);
```

In the structure of lpAddressStatus, LDK TSP fills dwDevSpecificSize, dwDevSpecificOffset and the corresponding data field. Especially, LDK TSP can return SMDR, All Station Status, All CO Line Status, Station Information, CO Line Information, and All Station' s Numbers. If a TAPI application want to get those data, those data receiving process must be started with **lineDevSpecific** function related to those data. Whenever those data occurs, the LINE_DEVSPECIFIC message is sent to the callback function. Then, the application can get

those data using this function. Refer to **lineDevSpecific** function (SMDR, All Station Status, All CO Line Status, Station Information, CO Line Information, and All Station' s Number),

lineGetCallInfo

The **lineGetCallInfo** function enables an application to obtain fixed information about the specified call.

```
LONG lineGetCallInfo(
   HCALL hCall,
   LPLINECALLINFO lpCallInfo
);
```

LDK TSP fills dwCallerID, dwCallerIDName, dwCalledID, dwCalledIDName, dwConnectedID, dwConnectedIDName when proper ID and Name is received. The dwRedirectingID and dwRedirectingIDName are provided to the ringing station when the DID incoming call is forwarded from a station to another station. The dwRedirectionID and dwRedirectionIDName are unavailable.

lineMakeCall

The **lineMakeCall** function places a call on the specified line to the specified destination address. Optionally, call parameters can be specified if anything but default call setup parameters are requested. The function is completed asynchronously.

```
LONG lineMakeCall(
   HLINE hLine,
   LPHCALL lphCall,
   LPCSTR lpszDestAddress,
   DWORD dwCountryCode,
   LPLINECALLPARAMS const lpCallParams
);
```

LDK TSP supports the following functions:

- **CO Call**

  " Cxx…x" or " @xx…x" in the lpszDestAddress field where xs are 0 through 9,*, and #. The front three xs represent CO line number and the following digits are real dialing numbers.

- **CO Group Call**

  " Gxx…x" or " $xx…x" in the lpszDestAddress field where xs are 0 through 9, *, and #. The front three xs represent CO line group number and the following digits are real dialing

numbers.

- **Networking Call**

  " xxx" in the lpszDestAddress field where xxx is extension number or station group number.

- **Outside/Long distance Call**

  " ss…sxx…x" in the lpszDestAddress field where ss…s is the Outside Access or Long distance Access string you have described at Telephony of Control Panel in your computer, and the following digits are real dialing number. The same response, that comes out when you Dial 9(0 in some countries) and the following digits in your keyset, will come out.

LDK TSP ignores the value in the dwCountryCode field. Also, the following four special functions are supported:

- **Station Speed Dial**

  " SPD/xx…x" in the lpszDestAddress field where xx…x is:

  1. " Cyyyzzz" or " @yyyzzz" where yyy is CO number and zzz is speed dial number.
  2. " Gyyyzzz" or " $yyyzzz" where yyy is CO group number and zzz is speed dial number.
  3. " zzz" where zzz is speed dial number.(000 ~ 099)

- **System Speed Dial**

  " SPD/xx" in the lpszDestAddress field where x is:

  1. " Cyyyzzzz" or " @yyyzzzz" where yyy is CO line number and zzzz is system speed dial number.
  2. " Gyyyzzzz" or " $yyyzzzz" where yyy is CO group number and zzzz is system speed dial number.
  3. " zzzz" where zzzz is system speed dial number.(2000 ~ 4999)

- **Last Number Redial(LNR)**

  " LNR/xx…x" in the lpszDestAddress field where x is:

  1. " Cyyy" or " @yyy" where yyy is CO line number.
  2. " Gyyy" or " $yyy" where yyy is CO group number.
  3. (None)

- **Save Number Redial(SNR)**

  " **SNR/xx…x**" in the lpszDestAddress field where x is:

  1. " Cyyy" or " @yyy" where yyy is CO line number.
  2. " Gyyy" or " $yyy" where yyy is CO group number.
  3. (None)

- **Make a call to a Conference Room**

  " **xxx/yyyyy**" in the lpszDestAddress field where x is:

  1. " xxx" is a Conference Room number.
  2. " yyyyy" is a password.

lineNegotiateAPIVersion

The **lineNegotiateAPIVersion** function allows an application to negotiate an API version to use.

```
LONG lineNegotiateAPIVersion(
   HLINEAPP hLineApp,
   DWORD dwDeviceID,
   DWORD dwAPILowVersion,
   DWORD dwAPIHighVersion,
   LPDWORD lpdwAPIVersion,
   LPLINEEXTENSIONID lpExtensionID
);
```

The negotiated API version number of LDK TSP with the pointer lpdwAPIVersion is

Low Version: 0x00010004

High Version: 0x00020001

The extension identifiers of LDK TSP in the structure pointed by lpExtensionID are

lpExtensionID->dwExtensionID0: 0x0ccb18c0

lpExtensionID->dwExtensionID1: 0x10202109

lpExtensionID->dwExtensionID2: 0x80002f8d

lpExtensionID->dwExtensionID3: 0xf354241e

lineNegotiateExtVersion

The **lineNegotiateExtVersion** function allows an application to negotiate an extension version to use with the specified line device. This operation need not be called if the application does not support extensions.

```
LONG lineNegotiateExtVersion(
   HLINEAPP hLineApp,
   DWORD dwDeviceID,
   DWORD dwAPIVersion,
   DWORD dwExtLowVersion,
   DWORD dwExtHighVersion,
   LPDWORD lpdwExtVersion
);
```

The negotiated extension version of LDK TSP with the pointer lpdwExtVersion is

Low Version: 0x00020001

High Version: 0x00020001

linePark

The **linePark** function parks the specified call according to the specified park mode. The function is completed asynchronously.

```
LONG linePark(
   HCALL hCall,
   DWORD dwParkMode,
   LPCSTR lpszDirAddress,
   LPVARSTRING lpNonDirAddress
);
```

LINEPARKMODE_NONDIRECTED in the field dwParkMode is applied only in 3rd party mode.

lineRedirect

The **lineRedirect** function redirects the specified offering call to the specified destination address.

```
LONG lineRedirect(
   HCALL hCall,
   LPCSTR lpszDestAddress
   DWORD dwCountryCode
);
```

If the offering call is external call, LDK TSP can support only the redirection of DID call.

lineSetStatusMessages

The **lineSetStatusMessages** function enables an application to specify which notification messages to receive for events related to status changes for the specified line or any of its

addresses.

```
LONG lineSetStatusMessages(
   HLINE hLine,
   DWORD dwLineStates,
   DWORD dwAddressStates
);
```

If this function is executed, you will receive the following messages:

- The number of Message Wait.
- The number of voice message
- Forward information
- The range of conference room (IP LDK or later version)

8.2 Phone Device Functions

The TAPI phone-device class functions supported by LDK TSP are described in the following table:

Function	LDK TSP Support
phoneClose	Supported.
phoneDevSpecific	Supported.
phoneGetData	Supported.
phoneGetDevCaps	Supported.
phoneGetID	Supported.
phoneInitializeEx	Supported.
phoneNegotiateAPIVersion	Supported.
phoneNegotiateExtVersion	Supported.
phoneOpen	Supported.
phoneShutdown	Supported.

phoneDevSpecific

The **phoneDevSpecific** function is used as a general extension mechanism to enable a Telephony API implementation to provide features not described in the other TAPI functions. The meanings of these extensions are device specific.

```
LONG phoneDevSpecific(
   HPHONE hPhone,
   LPVOID lpParams,
   DWORD dwSize
);
```

LDK TSP supports the following device-specific functions:

- **Wake-Up**

  " WU/xx…x" in the lpParams field where xx…x is:

  1. HHMM (HH:hour, MM:minute)
  2. HHMM/S (S:once)
  3. HHMM/C (C:continuous)
  4. CURR (current): to get current wake up time. You will get the " none" string if current wake up time does not be set.
  5. DELE (delete): to delete current wake up time.

- **Authorization Code/Password**

  " ACP/xx…x" in the lpParams field where xx…x is:

  1. yyyyy/zzzzz (yyyyy : old password, zzzzz : new password) where each y and z is 0 through 9,*, and #.
  2. NEWPW/yyyyy (yyyyy : new password) where y is 0 through 9,*, and #.
  3. CURPW: to get current password. If there is no current password, LDK system returns " NOPAS."

- **Temporal Station COS Change**

  " TSCC/xx…x" in the lpParams field where xx…x is:

  1. D (down)
  2. C (current)
  3. R/yyyyy where yyyyy is password

- **Preselected Message Program**

  " PMP/xx…x" in the lpParams field where xx…x is:

  1. A message string that can contain alphabet, :, (,), 0 through 9, or space character within 24 letters.
  2. /CUR (current message), PMP//CUR
  3. /DEL (delete message), PMP//DEL

- **ICM Answer Mode**

  " IAM/x" in the lpParams field where x is:

  1. H (hands free mode)
  2. T (tone ring mode)
  3. P (privacy mode)
  4. C (current)

- **BGM Assignment**

  " BGM/xx" in the lpParams field where x is:

  1. 00 (No use)
  2. 01 (BGM Channel 1)
  3. 02 (BGM Channel 2)
  4. 03 (BGM Channel 3)
  5. 04 (BGM Channel 4)
  6. 05 (BGM Channel 5)
  7. 06 (BGM Channel 6)
  8. 07 (BGM Channel 7)

9. 08 (BGM Channel 8)
10. 09 (BGM Channel 9)
11. 10 (BGM Channel 10)
12. 11 (BGM Channel 11)
13. 12 (BGM Channel 12)
14. Cu(Current BGM Channel)

- **Preselected Message Selection**

  " PMS/mm/xx…x" in the lpParams field where mm is:

  1. 00: Display preselected message programmed in each station. " /xx…x" must not be set.
  2. 01: Display " LUNCH / RETURN AT hh:mm" where xx…x is hhmm.
  3. 02: Display " ON VACATION / RETURN AT dd mon" where xx…x is ddmm.
  4. 03: Display " OUT OF OFFICE / RETURN AT hh:mm" where xx…x is hhmm.
  5. 04: Display " OUT OF OFFICE / RETURN AT dd mon" where xx…x is mmdd.
  6. 05: Display " OUT OF OFFICE / RETURN UNKNOWN" where /xx…x must not be set.
  7. 06: Display " CALL : telephone number" where xx…x is telephone number (Max: 18 digits).
  8. 07: Display " IN OFFICE : STA ssss" where xx…x is ssss.
  9. 08: Display " IN A MEETING / RETURN TIME hh:mm" where xx…x is hhmm.
  10. 09: Display " AT HOME." " /xx…x" must not be set.
  11. 10: Display " AT BRANCH OFFICE" where /xx…x must not be set.
  12. 11 – 20: Display preselected messages programmed in attendant..
  13. 21: Current preselected message where /xx…x must not be set.
  14. 99: Unset current preselected message.

- **Differential Ring** (Digital Keyset Only)

  " DR/x" in the lpParams field where x is:

  1. 1 ~ 4 (Ring Type)
  2. 0 (Current Ring Type)

- **LCD Language Mode**

  " LLM/x" in the lpParams field where x is:

  1. 0 (English)
  2. 1 (Corresponding Country)
  3. 2 (Current)

- **Headset Mode Program**

  " HMP/x" in the lpParams field where x is:

  1. 0 (Headset mode off)
  2. 1 (Headset mode on)
  3. 2 (Current)

phoneGetData

The **phoneGetData** function uploads the information from the specified location in the open phone device to the specified buffer.

```
LONG phoneGetData(
  HPHONE hPhone,
  DWORD dwDataID,
  LPVOID lpData,
  DWORD dwSize
);
```

LDK TSP supports the following functions:

- **Wake-Up**

  " 0x01" in the dwDataID field

  The lpData field and its meaning are:

  1. DELET: No setting
  2. HHMMF where HH:hour, MM:minute, F: C(continuous)/S(single)

- **Authorization Code/Password**

  " 0x02" in the dwDataID field

  The lpData field and its meaning are:

  1. NOPAS: No password
  2. xxxxx: Current password

- **Temporal Station COS Change**

  " 0x03" in the dwDataID field

  The lpData field and its meaning are:

  1. D: COS Down
  2. R: COS Restored

- **Preselected Message Program**

  " 0x04" in the dwDataID field

  The lpData field and its meaning are:

  1. xx…x: null terminated string

- **ICM Answer Mode**

  " 0x05" in the dwDataID field

  The lpData field and its meaning are:

  1. H: hands free
  2. T: Tone ring
  3. P: Privacy

- **BGM Assignment**

  " 0x06" in the dwDataID field

  The lpData field and its meaning are:

  1. 0: not in use
  2. x: channel no(1 or 2)

- **Preselected Message Selection**

  " 0x07" in the dwDataID field

  The lpData field and its meaning are:

  1. /mm/xx…x where mm is message number and xx…x is the related string. Refer to **phoneDevSpecific** function

- **Differential Ring** (Digital Keyphone Only)

  " 0x08" in the dwDataID field

  The lpData field and its meaning are:

  1. x: ring type(1-4)

- **LCD Language Mode**

  " 0x09" in the dwDataID field

  The lpData field and its meaning are:

  1. 0: English
  2. 1: Domestic

- **Headset/Speakerphone Mode Program**

  " 0x0B" in the dwDataID field

  The lpData field and its meaning are:

  1. 0: Off
  2. 1: On

phoneNegotiateAPIVersion

The **phoneNegotiateAPIVersion** allows an application to negotiate an API version to use for the specified phone device.

```
LONG phoneNegotiateAPIVersion(
   HPHONEAPP hPhoneApp,
   DWORD dwDeviceID,
   DWORD dwAPILowVersion,
   DWORD dwAPIHighVersion,
   LPDWORD lpdwAPIVersion,
   LPPHONEEXTENSIONID lpExtensionID
);
```

The negotiated API version number of LDK TSP with the pointer lpdwAPIVersion is

Low Version: 0x00010004

High Version: 0x00020001

The extension identifiers of LDK TSP in the structure pointed by lpExtensionID are

lpExtensionID->dwExtensionID0: 0x7bdc3120

lpExtensionID->dwExtensionID1: 0x10204a78

lpExtensionID->dwExtensionID2: 0x8000318d

lpExtensionID->dwExtensionID3: 0xf354241e

phoneNegotiateExtVersion

The **phoneNegotiateExtVersion** function allows an application to negotiate an extension version to use with the specified phone device. This operation need not be called if the application does not support extensions.

```
LONG phoneNegotiateExtVersion(
   HPHONEAPP hPhoneApp,
   DWORD dwDeviceID,
   DWORD dwAPIVersion,
   DWORD dwExtLowVersion,
   DWORD dwExtHighVersion,
   LPDWORD lpdwExtVersion
);
```

The negotiated extension version of LDK TSP with the pointer lpdwExtVersion is

Low Version: 0x00020001

High Version: 0x00020001

8.3 Device Specific Messages

LINE_DEVSPECIFIC

The **LINE_DEVSPECIFIC** message is sent to notify the application about device-specific events occurring on a line, address, or call. The meaning of the message and the interpretation of the parameters are described below.

LINE_DEVSPECIFIC
dwDevice **= (DWORD) hLineOrCall;**
dwCallbackInstance **= (DWORD) hCallback;**
dwParam1 **= (DWORD) DeviceSpecific1;**
dwParam2 **= (DWORD) DeviceSpecific2;**
dwParam3 **= (DWORD) DeviceSpecific3;**

LDK TSP sends this message to application to notify the following events:

- **SMDR, All Station Status, All CO Line Status have received.** (Refer to **lineDevSpecific** function)

  hLine in the dwDevice field,

  " 0x01" in the dwParam1 field.

  1. SMDR data is received.
     " 0x01" in the dwParam2 field.
  2. All Station Status data is received.
     " 0x04" in the dwParam2 field.
  3. All CO Line Status data is received.
     " 0x05" in the dwParam2 field.
  4. All Station Numbers are received..
     " 0x08" in the dwParam2 field.
  5. Station Information is received.
     " 0x09" in the dwParam2 field.
  6. CO line Information is received.
     " 0x0A" in the dwParam2 field.
  7. All Park Numbers are received.
     " 0x0B" in the dwParam2 field.

After receiving this message, the application can get the data using **lineGetAddressStatus** function.

***Note**: 3rd party mode only*

- **The UserUserInfo data have been accumulated over 2.**

  hCall in the dwDevice field,

  " 0x02" in the dwParam1 field,

  " xx" in the dwParam2 field where xx is the number of accumulated **UserUserInfo** data.

- **The response of " Get System Time" in lineDevSpecific function has received.**

  hLine in the dwDevice field,

  " 0x03" in the dwParam1 field,

  " yyyymmmm" in the dwParam2 field where yyyy is year, mmmm is month,

  " ddhhmmss" in the dwParam2 field where dd is date, hh is hour, mm is mimute, ss is second.

- **The MSGWait (ICM number) has received.**

  hLine in the dwDevice field,

  " 0x04" in the dwParam1 field,

  " x" in the dwParam2 field where x is 0x01(ON) or 0x00(OFF),

  " y" in the dwParam3 field where y is the station number which leave the waiting message (DWORD).

- **DVU MSG has received.**

  hLine in the dwDevice field,

  " 0x05" in the dwParam1 field,

  " 0x01" in the dwParam2 field,

  " x" in the dwParam3 field where x is received DVU message number.

- **DND state has changed.**

  hLine in the dwDevice field,

  " 0x06" in the dwParam1 field,

  " 0x01" in the dwParam2 field,

  " x" in the dwParam3 field where x is 0x01(ON) or 0x00(OFF).

- **The station' s state has changed to DVU Forwarded.**

  hLine in the dwDevice field,

  " 0x06" in the dwParam1 field,

  " 0x02" in the dwParam2 field.

- **Station number related to physical number is sent.**

  hLine in the dwDevice field,

  " 0x07" in the dwParam1 field,

  " x" in the dwParam2 field where x is the physical number of the station in DWORD, which has described in **lineDevSpecific** function,

  " y" in the dwParam3 field where y is the station number of the station, x, in 4-byte characters.

  ***Note**: 3rd party mode only*

- **Station Status has changed.**

  hLine in the dwDevice field,

  " 0x08" in the dwParam1 field,

  " x" in the dwParam2 field where x is the physical number of the station in DWORD,

  " y" in the dwParam3 field where y is the changed status of the station. (Status is described in 0 Device Specific Data Format)

  ***Note**: 3rd party mode only*

- **CO Line Status has changed.**

  hLine in the dwDevice field,

  " 0x09" in the dwParam1 field,

  " x" in the dwParam2 field where x is the physical number of the CO line in DWORD,

  " y" in the dwParam3 field where y is the changed status of the CO line. (Status is described in 0 Device Specific Data Format)

  ***Note**: 3rd party mode only*

- **My Station Status has changed.(**Opened line' s status has changed)

  hLine in the dwDevice field,

  " 0x0B" in the dwParam1 field,

  " x" in the dwParam2 field where x is the changed status of my station in DWORD, (Status is described in 0 Device Specific Data Format)

  ***Note**: 3rd party mode only*

- **Wake up ring**

  hLine in the dwDevice field,

  " 0x10" in the dwParam1 field,

  " x" in the dwParam2 field where x is 0x01(Ring) or 0x00(Stop),

- **Linked Pair**

  hLine in the dwDevice field,

  " 0x11" in the dwParam1 field,

  " x" in the dwParam2 field where x is 0x01(Slave is in use) or 0x00(Stop),

  Note: 1. The master receives CTI Message, but the slave doesn't.

  2. When the slave is in use, the master can't use CTI Functions

- **Park Number**

  hLine in the dwDevice field,

  " 0x12" in the dwParam1 field,

  " x" in the dwParam2 field where x is the Park number.

  " y" in the dwParam3 field where y is the CO line.

- **Station number has changed**

  hLine in the dwDevice field,

  " 0x36" in the dwParam1 field,

  " x" in the dwParam2 field where x is the Port Number

  " y" in the dwParam3 field where y is the Changed Station number

- **Range of Conference Room**

  hLine in the dwDevice field,

  " 0x37" in the dwParam1 field,

  " x" in the dwParam2 field where x is the Start Number

  " y" in the dwParam3 field where y is the Last number

- **Dummy Station**

  hLine in the dwDevice field,

  " 0x38" in the dwParam1 field,

  " x" in the dwParam2 field where x is 0x00(Normal Keyset) or 0x01(Dummy Keyset).

  " y" in the dwParam3 field where y is 0x00(Logon State) or 0x01(Dummy State).

- **UCD DND has changed**

  hLine in the dwDevice field,

  " 0x39" in the dwParam1 field,

  " x" in the dwParam2 field where x is 0x00(OFF) or 0x01(ON).

  " y" in the dwParam3 field where x is a UCD group number.

- **The Keyset has restarted**

  hLine in the dwDevice field,

  " 0xFE" in the dwParam1 field,

  " 0x01" in the dwParam2 field.

- **The Keyset line has been disconnected**

  hLine in the dwDevice field,

  " 0xFE" in the dwParam1 field,

  " 0x03" in the dwParam2 field.

Device Specific Data Format

If an application requests SMDR, All Station Status, All CO Line Status for the line device using **lineDevSpecific** function, LDK TSP gathers the data and sends the application message of each data. Then, the application can obtain the data using **lineGetAddressStatus**. In the LINEADDRESSSTATUS structure of the function, LDK TSP fills dwDevSpecificSize, dwDevSpecificOffset, and attaches the corresponding data, which is represented as following. (Refer to **lineGetAddressStatus** in this document and in *Microsoft Platform SDK*)

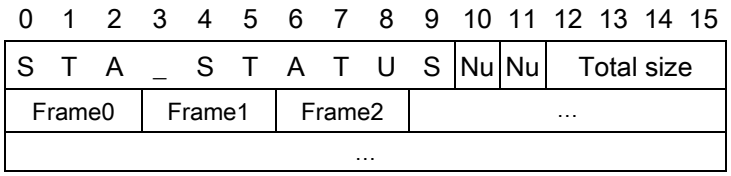
- **SMDR Data Format**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14 15 16 17
S	M	D	R	_	D	A	T	A	Nu	Nu	Nu	Nu	Nu	Total size
FrameNum0				FrameSize0				FrameData0						
FrameData0(unfixed size)														
FrameNum1				FrameSize1				FrameData1						
FrameData1(unfixed size)														

- SMDR_DATA : ascii string, “ SMDR_DATA” (9 bytes)
- Nu : NULL character
- TotalSize : the total size of SMDR DATA format (4 bytes)
- FrameNum : the number of FrameData (4 bytes)
- FrameSize : the size of FrameData (4 bytes)
- FrameData : SMDR data (unfixed size). Refer to the following table

Field Name	No. of bytes	Descriptions
Sequential number	4	Sequential number (range 0000 ~ 9999)
Space	1	
Station	5	Station number (2, 3, or 4 digits) CO number (CO xxx)
Space	1	
CO	3	CO Line number
Time	8	Call time serviced, 01:10:05(HH:MM:SS)
Space	1	
Start time	14	Start time, 25/11/01 14:20
Space	1	
Record type	1	t : Incoming CO transfer I : incoming CO (answered) O : outgoing CO T : outgoing CO transfer H : hold lost R : ring lost G : Group call lost
Dialed digits	18	Dialed digits (string)
Space	2	
Account group	2	Account Group Number field.
Space	1	
Metering count	5	Call metering count (record type : O, T)
Space	1	
Call cost	11	Call Cost (record type : O, T)
Space	1	
Account code	12	Account code field.

- **Station Status Data Format**



- STA_STATUS : ascii string, “ STA_STATUS” (10 bytes)
- Nu : NULL character
- Total size : total size of Station Status Data Format in byte (4 bytes)

- Frame0, Frame1, … $Frame_{Max}$: All Station Status Data frame (3 bytes)

0 - 1	2
X	Y

X : physical number of station (physical number is order of connection of the station to LDK system)

Y : one of the following status values

	Status	Hexa value
Station Status	ST_INVALID	0x01
	ST_HOLD	0x11
	ST_RING	0x12
	STEXT_IDLE	0x40
	STEXT_IDLE_WITH_DND	0x41
	STEXT_IDLE_WITH_RING	Not Support
	STEXT_ICM	0x43
	STEXT_COL	0x44
	STEXT_PGM	0x45
	ST_BUSY	0x10

- **CO Line Status Data Format**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C	O	L	_	S	T	A	T	U	S	Nu	Nu	Total size			
Frame0			Frame1			Frame2				…					
…															

- COL_STATUS : ascii string, “ COL_STATUS” (10 bytes)
- Nu : NULL character
- Total size : total size of CO Line Status Data Format in byte (4 bytes)
- Frame0, Frame1, … $Frame_{Max}$: All CO Line Status Data frame (3 bytes)

0 - 1	2
X	Y

X : physical number of CO line (physical number is order of connection of the CO line to LDK system)

Y : one of the following status values

	Status	Hexa value
CO Line Status	ST_INVALID	0x01
	ST_HOLD	0x11
	ST_RING	0x12
	STTRK_IDLE	0x80
	STTRK_HOLD_RECALL	0x81
	STTRK_HOLD_ATDRECALL	0x82
	STTRK_IDLE_WITH_DISABLE	Not Support
	ST_BUSY	0x10

- **All Station' s Logical Number Data Format**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S	T	A	_	N	U	M	B	E	R	Nu	Nu	Total Size			
Nu_0		Nu_1		Nu_2		…									
…										Nu_n					

- STA_NUMBER : ascii string, " STA_NUMBER" (10 bytes)
- Nu : NULL character
- Total size : total size of All Station' s Logical Number Data Format in bytes (4 bytes)
- Nu_0 : logical(assigned internal) number of the first station in integer(2 byte)
- Nu_1 : logical(assigned internal) number of the second station in integer(2 byte)
- Nu_n : logical(assigned internal) number of the *n* th station in integer(2 byte)

- **Station Information Data Format**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S	T	A	_	I	N	F	O	Nu	Nu	Nu	Nu	Total Size			
Pn		Sta Number				Sl	St	At	Ap			Ns	Sta Name ~		

- STA_INFO : ascii string, " STA_INFO" (8 bytes)
- Nu : NULL character
- Total size : total size of Station Information Data Format in byte (4 bytes)
- Pn : physical number of the station (2 byte)
- Sta Number : the number of the station in DWORD (4-byte integer)

  (ex> 0x00000064 : station 100)
- Sl : slot number of the station in LDK system (1 byte)
- St : status of the station. Refer to the Station Status value (1 byte)
- At : type of the associated device (1 byte)

  0x01 : the station is connected to a station

0x02 : the station is connected to a CO line

0x00/0xFF : the station is not connected

- Ap : physical number of the associated device (2 byte)

  (ex> At = 0x02, Ap = 0x0001 : the station is now using CO line 01)
- Ns : size of the station’ s name (1 byte)
- Sta Name : name of the station (size of Ns)

- **CO Line Information Data Format**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C	O	L	_	I	N	F	O	Nu	Nu	Nu	Nu	Total Size			
Pn		St	At	Ap		Ns	Cs	CO Name							
CallerID															

- COL_INFO : ascii string, “ COL_INFO” (8 bytes)
- Nu : NULL character
- Total size : total size of CO Line Information Data Format in byte (4 bytes)
- Pn : physical number of the CO line (2 byte)
- St : status of the CO line. Refer to the CO Line Status value (1 byte)
- At : type of the associated device (1 byte)

  0x01 : the CO line is connected to a station

  0x02 : the CO line is connected to a CO line

  0x00/0xFF : the station is not connected
- Ap : physical number of the associated device (2 byte)

  (ex> At = 0x01, Ap = 0x0001 : the CO line is used by first station)
- Ns : size of the CO line name (1 byte)
- Cs : size of the CallerID of the CO line (1 byte)
- CO Name : name of the CO line (size of Ns)
- CallerID : CallerID of the CO line (size of Cs)

- **All Park Number Data Format**

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
A	L	L	_	P	A	R	K	_	N	U	M	Total Size			
Nu_0				Nu_1				Nu_2				…			
…								Nu_n							

- ALL_PARK_NUM : ascii string, “ ALL_PARK_NUM” (12 bytes)

- Total size : total size of ALL Park Number Data Format in byte (4 bytes)
- Nu_0 : The first park number(4 byte)
- Nu_1 : The second park number(4 byte)
- Nu_2 : The third park number(4 byte)

9. Troubleshooting

- Lines and phones can not be seen on the server with the **tapimgmt.msc** command

Cause: The connection between LDK system and LDK TSP does not work correctly.

Solution: Ensure the following:

- 3rd party lock key is set up correctly.
- After installing LDK TSP, you should restart your computer.
- Ensure the LDK TSP configurations.

- One or more client computers cannot see the telephony server.

Cause: The server cannot be reached on the network because it is not set up correctly.

Solution: Ensure the following:

- The client computers can successfully log on to a domain.
- The client computers can successfully see the server with the **ping** command.
- That the telephony server has been configured correctly.

- Client computers cannot see lines on the telephony server.

Cause: The telephony server has not been set up correctly, or users have not been authorized to access lines on the server.

Solution: When a TAPI-based program accesses lines on the telephony server, the user who is running the program process is first authenticated. For the user to see the lines, they must have been assigned to the user. Ensure that the server has been set up correctly and that the user is assigned lines on the server.

- A client user cannot see lines or phones on the telephony server ever though the server is set up correctly and the lines or phones have been assigned to the user.

Cause: The client computer has not been enabled to use the telephony server.

Solution: Use **tcmsetup** on the client computer to specify the telephony server.

- A client user cannot see a new line on the telephony server, even though the server administrator has assigned the user to the line.

Cause: When you assign a currently running client user to a line on the telephony server, the new

settings will not be available until the user restarts the telephony service on the client computer.
Solution: Stop all client TAPI applications on the client so that the telephony service will shut down. When the client applications restart, they will be able to see the newly allocated lines. If you cannot get TAPI to shut down, restart the client computer.

- A client logon ID must be used in English and digits.

- Edit the lmhosts.sam – No response or Long delay time between the TAPI server and a client.

If a subnet of client computer IP address and a subnet of TAPI server IP address are different, it takes long time to get the TAPI information between the TAPI server and a client.

The lmhosts.sam file is a static file that assists with remote NetBIOS name resolution on computers that cannot respond to NetBIOS name-query broadcasts. It contains NetBIOS name-to-IP addresses mappings.

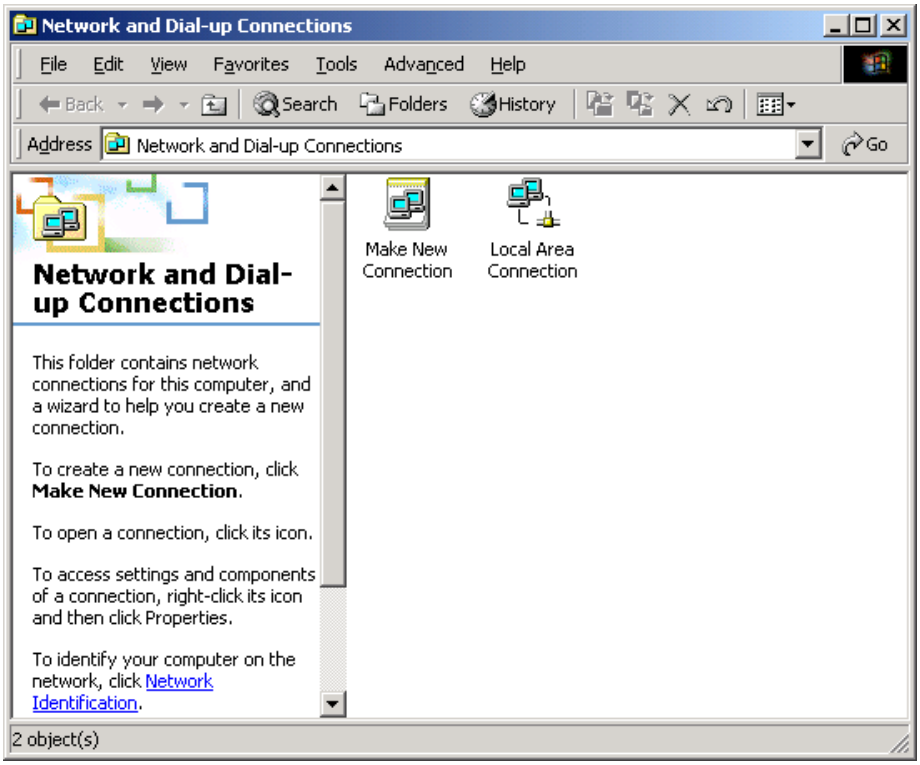
To edit the lmhosts.sam

1. Open lmhosts.sam

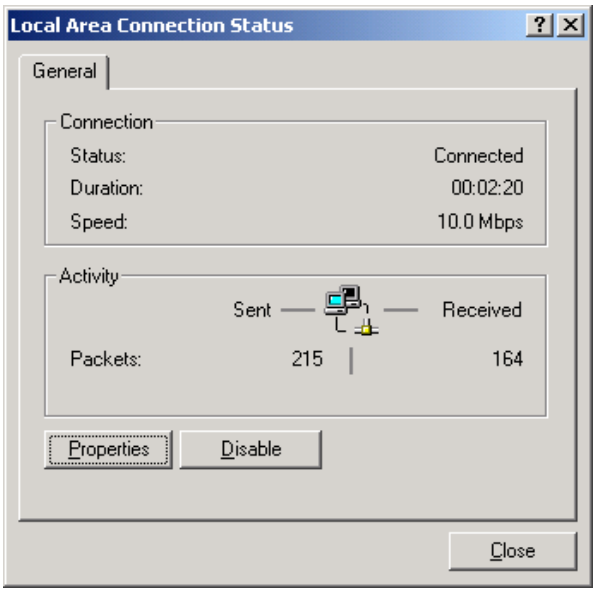
   Tipically, the lmhosts file is stored in the systemroot\System32\Drivers\Etc folder.
2. Add NetBIOS name-to-IP Address
3. Save the file.

To configure TCP/IP to use WINS

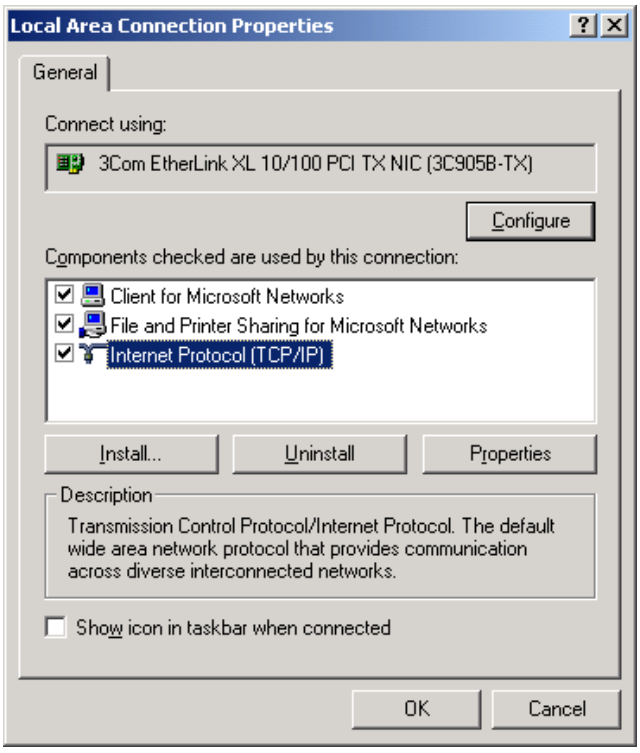
1. Open **Network and Dial-up Connections**



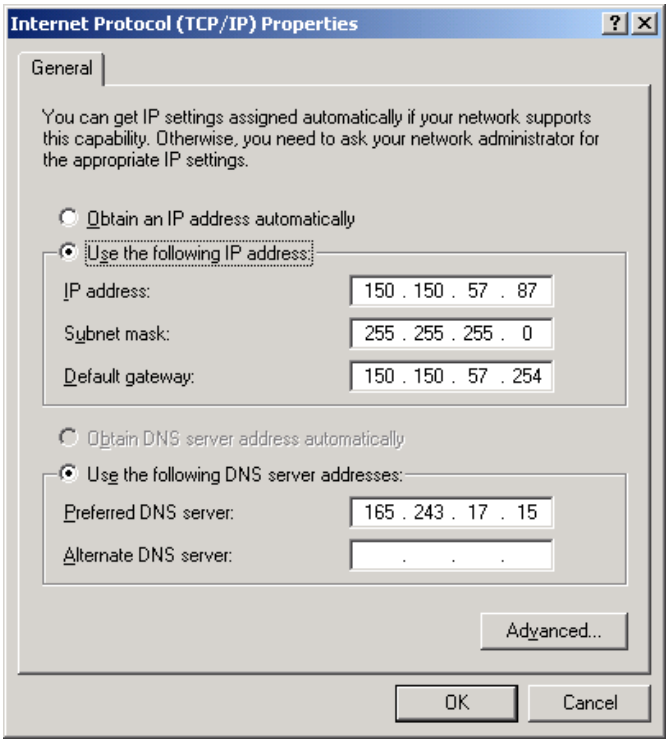
2. Right-click the network connection you want to configure, and then click **Properties.**

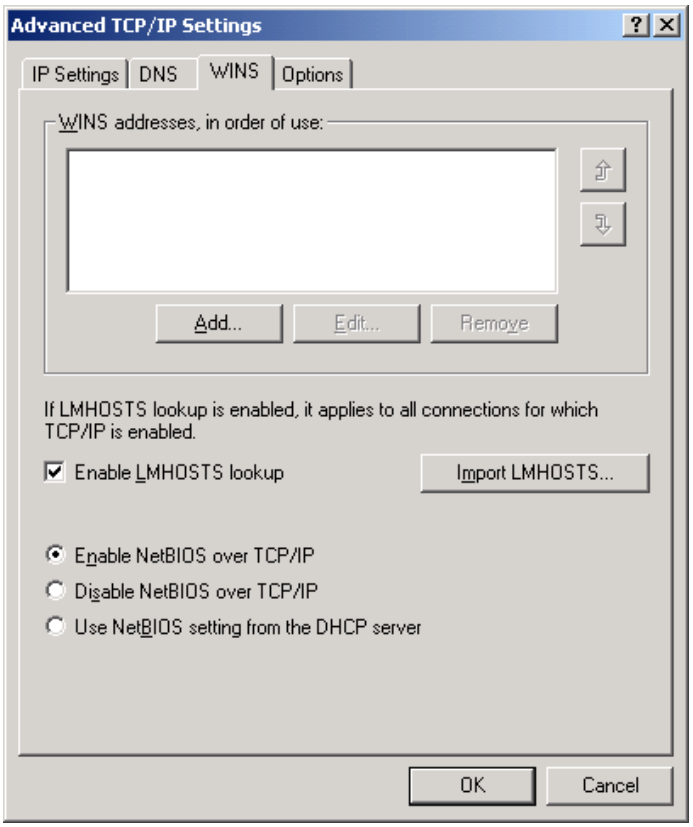


3. On the **General** tab (for a local area connection) or the **Networking** tab (all other connections), click **Internet Protocol (TCP/IP)**, and then click **Properties.**



4. Click **Advanced**, click the **WINS** tab.





To enable the use of the lmhosts file to resolve remote NetBIOS names, select the **Enable LMHOSTS lookup** check box. This option is enabled by default.

To specify the location of the file that you want to import into the lmhosts file, click **Import LMHOSTS**, and select the file in the **Open** dialog box.