

CTI32 Guide

This document will describe the various documentation files. It will then give an overview of the examples and what features are shown in each example. The final section will cover selected topics and frequently asked questions about the toolkit.

Documentation Files

All documentation files are located in the Documentation folder under the install folder.

GettingStarted – READ THIS FIRST. This gives an overview of the product. It describes the various modules and major parts to CTI32. It describes the CTI32 engine and how it works. It describes the process of how an incoming and outgoing call works. It describes how to get the inbound call sample running. It describes the files installed on your system.

Release Notes – This file contains new features or bug fixes for the current and past releases.

CTI32Guide – This document.

CTI32Config – This document describes the CTI32Config utility and all of its options. It helps with configuring your system to work with the Engine. All the various parameters are described. How to view the log files and use TMonitor are also described.

CTI32NetLib Documentation – This is a compiled HTML help file that describes the CTI32NetLib (Telephony functions) uses for .NET applications.

CTI32Engine Documentation – This is a compiled HTML help file that describes the classes in the CTI32Engine.

CTI32.DLL – This is a PDF file that describes the capabilities of the low level CTI32.DLL. .NET programmers should not need this. If you are developing an application in C++ or Delphi and working directly with CTI32.DLL, then you will need this PDF file.

Sample Applications

The sample applications are located under the Samples Folder under the installation directory. They are organized under C# samples (CS) and VB samples. You can easily change your cti32engine.config file to run the sample by going into the CTI32config utility and pressing the sample button.

Note: The samples should compile fine as long as your installation folder was c:\program files\cti32. If you installed to a different folder, you will have to go into your solution explorer and under References delete CTI32Engine and CTI32NetLib. Re-add these references by right clicking on References and selecting "Add Reference". Select the Projects tab and press the Browse button. Navigate to the installation folder and select CTI32Engine.DLL and CTI32NetLib.DLL.

InboundIVR – Demonstrates how to write a simple inbound IVR application. This sample also demonstrates the following:

- Logging to the CTI32Engine.Log file
- Displaying any information on the TMonitor screen (Note: CTI32 allows you to replace TMonitor with your own application)
- How to Play a File
- How to Get Digits
- How to Record a File
- How to play back numbers
- How to get the DNIS (Dialed Number)
- How to get the ANI (Callers Number)
- How to trap for the caller hanging up

OutboundDialer – Demonstrates how to write an outbound application. This sample also demonstrates the following:

- How to read an XML configuration file using the CTI32 tools
- How to use the Dispatch Thread to trigger outbound phone calls
- How to have CTI32 select an available line and initiate the dial
- How to choose the line you wish to dial out on and then initiate the call
- How to Dial using Call Progress
- How to "spoofer" the outbound callerID on PRI Lines
- How to determine if a Human or Machine answered the phone
- How to leave a message on an answering machine (wait for silence before playing a file)
- How to use Global Tone to check to see if you reached a fax machine (sometimes the fax detection in the PerfectCall ain't so perfect)

This application reads a list of phone numbers from the OutboundDialer.config file and dials them. If a HUMAN answers it plays one message or if a machine answers, it plays a different file.

Conference - Demonstrates how place calls into a conference. This sample also demonstrates the following:

- Logging to the CTI32Engine.Log file
- Displaying any information on the TMonitor screen (Note: CTI32 allows you to replace TMonitor with your own application)
- How to Play a File

- How to use the Dispatch Thread for Initialization
- Entering And Leaving Conferences
- Simple Conference Management

BridgedCall - Demonstrates how accept a call, dial out and connect the two parties together.

This sample also demonstrates the following:

- Logging to the CTI32Engine.Log file
- Displaying any information on the TMonitor screen (Note: CTI32 allows you to replace TMonitor with your own application)
- How to Play a File
- How to Get Digits
- How to play back numbers
- How to trap for the caller hanging up
- How to execute a two B-Channel Transfer (TBCT)

MakeRecordings – This is a convenience sample so that you can easily play and record Wave files that you can use in your applications. Simply switch to this sample, call in, and record or play files in the format #.WAV. (you enter the number)

This sample also demonstrates the following:

- How easy it is to create a useful application
- How to Play a File
- How to Get Digits
- How to Record a File
- How to trap for the caller hanging up

IncomingFax – This demo application will receive a fax and store it as a multi-page TIF file. Upon successful receipt, it will immediately dial back to the phone number received as the CSID and send the fax back. You should have no dashes in your CSID. It should be in the format 3034799834.

This sample also demonstrates the following:

- How easy it is to create a useful application
- How to receive a fax
- How to send a fax
- How to Route for a fax
- How to store a fax image
- How to setup a fax callback. You should get events on each page break.

NOTE: CTI32 now supports two fax API's. GammaLink API (GDK) and the Dialogic R4 Fax API. The default is the GDK. To use the R4 Fax API you must change CTI32.INI file. Set it as follows:

```
[FAX]
FAX_API=1
```

The 1 means the R4 API.

CallingCard – Demonstrates how easy it is to write a useful application – A Calling Card application. The user dials in, enters a PIN, enters a phone number to dial, then the call is bridged together. In addition to all the features demonstrated in the InboundIVR and the OutboundDialer, this application also demonstrates:

- How to Route a call together
- How to read from a Database

The calling card Sample can run with or without a database.

To run with a Database you must perform the following steps:

1. Create a table in your ODBC compatible database called Customer. There must be a column called Pin.
2. Create a Database DSN (Control Panel / Administrative Tools / Data Sources (ODBC))
3. Add a connect string to the CTI32Config / DatabaseConnectionString (i.e. DSN=DSNName)
4. Save the config
5. Restart the service

See section below “Database Access from your IVR” for more information on Database access.

Selected Topics

Database Access from your IVR

Certainly you can add your own database access routines in your application. If you are using pooled connections, connecting each time you wish to access the database similar to the way you would in web forms can even be relatively efficient.

However, CTI32 has implemented an easy to use Database class for you to use. It automatically makes one connection to the database when the engine starts. Each access to the database is funneled through this connection no matter which channel (thread) it is being called from. This will be the most efficient form of database access possible. CTI32 will handle all the synchronization between concurrent accesses all automatically.

Note: you should have your application read a table once per hour so that the database will not close this connection due to inactivity.

Database access is set up to work through ODBC. The DbConnection class does have the capability to work with the native SQLServer connection type. However, this is not covered here. Please contact technical support if you would like to use the native SQLServer connection type.

To set up database access perform the following:

1. Set up your ODBC Dataset name (DSN) on your computer. You do this through the Windows “Administrator Tools” – Data Sources (ODBC). Usually you will want to set up the DSN under the “System DSN” tab and not the “User DSN” tab, although it might work either way.
2. Enter the connection string in the “configuration utility” under the “DatabaseConnectionString”. Some examples may be:
 - a. dsn=IvrDB;UID=sa;PWD=secret
 - b. dsn=Dialer
3. If your DatabaseConnectionString has something in it, the engine will automatically create a DbConnection for you and Connect to the database. You can review the CTI32engine.log to see if this was successful. It will store the connection in cfg.dbConnection. The engine will also close the connection when you stop the engine.
4. In your dispatch thread, your incomingCall thread, or your OutgoingCall thread, you can easily use this connection by creating a Db class as follows:
 - a. Db db = new Db(cfg.log,cfg.dbConnection)
 - b. db.port=lineData.port (used for logging to see which port was doing what)

Select statements would look like this:

```
sqlStmt=String.Format("SELECT * FROM JobQueue WHERE jobNumber = '{0}'", currJob);
dr=db.GetFirst(sqlStmt);
if(dr == null)
{
    log.Write("Could not find JobNumber");
}
```

An Update statement would look like this:

```
sqlStmt=String.Format("UPDATE SystemSettings SET currentJob = '{0}' WHERE systemSettingsId = 1", currJob);
ret=db.Update(sqlStmt);
```

These are just the basics. There are many other features that you can refer to in the CTI32Engine documentation.

Also keep checking for new example programs. We will soon have a database access example in C# and VB.NET for you to look at.

System VAP File

A VAP file is kind of a zip file for voice files. It allows you to put a whole bunch of small voice files into one file each having a numerical index. VAP files had its origin in the early days of Dialogic development.

CTI32 has defined its own VAP structure for playing of Numbers, Dates, Times, Ordinals, etc. Included in the install is the English.VAP file which has the basic files in it to support these functions. You may want to re-create this file in your own branded voice. This section describes how to create such a file.

CTI32 supports both English and Spanish System VAP files for playing numbers, dates, times etc.

Each segment in the VAP file has a number identifying it. For the CTI32 methods to work correctly, the right recording must be in the right index. Note: All recordings in a VAP must be in the same voice format. The English.VAP and Spanish.VAP file is recorded in 11Khz PCM Mono 8-bit WAVE.

The following dictionary describes the voice locations needed by CTI32. Of course you can add your own to this as long as you don't use the same numbers as what CTI32 needs. (start at numbers over 1000)

	Numbers	Numeros
0	Zero	Cero
1	One	Uno
2	Two	Dos
3	Three	Tres
4	Four	Cuatro
5	Five	Cinco
6	Six	Seis
7	Seven	Siete
8	Eight	Ocho
9	Nine	Nueve
10	Ten	Diez
11	Eleven	Once
12	Twelve	Doce
13	Thirteen	Trece
14	Fourteen	Catorce
15	Fifteen	Quince
16	Sixteen	dieciséis
17	Seventeen	Diecisiete
18	Eighteen	Dieciocho
19	Nineteen	Diecinueve
20	Twenty	veinte
21	Twenty-one	veintiuno
22	Twenty-two	veintidós
23	Twenty-Three	veintitrés
24	Twenty-Four	veinticuatro
25	Twenty-five	veinticinco
26	twenty six	veintiséis
27	twenty seven	veintisiete
28	twenty eight	veintiocho
29	twenty nine	veintinueve
30	thirty	treinta
31	thirty one	treinta y uno

32	thirty two	treinta y dos
33	thirty three	treinta y tres
34	thirty four	treinta y cuatro
35	thirty five	treinta y cinco
36	thirty six	treinta y seis
37	thirty seven	treinta y siete
38	thirty eight	treinta y ocho
39	thirty nine	treinta y nueve
40	forty	cuarenta
41	forty one	cuarenta y uno
42	forty two	cuarenta y dos
43	forty three	cuarenta y tres
44	forty four	cuarenta y cuatro
45	forty five	cuarenta y cinco
46	forty six	cuarenta y seis
47	forty seven	cuarenta y siete
48	forty eight	cuarenta y ocho
49	forty nine	cuarenta y nueve
50	fifty	cincuenta
51	fifty one	cincuenta y uno
52	fifty two	cincuenta y dos
53	fifty three	cincuenta y tres
54	fifty four	cincuenta y cuatro
55	fifty five	cincuenta y cinco
56	fifty six	cincuenta y seis
57	fifty seven	cincuenta y siete
58	fifty eight	cincuenta y ocho
59	fifty nine	cincuenta y nueve
60	sixty	sesenta
61	sixty one	sesenta y uno
62	sixty two	sesenta y dos
63	sixty three	sesenta y tres
64	sixty four	sesenta y cuatro
65	sixty five	sesenta y cinco
66	sixty six	sesenta y seis
67	sixty seven	sesenta y siete
68	sixty eight	sesenta y ocho
69	sixty nine	sesenta y nueve
70	seventy	setenta
71	seventy one	setenta y uno
72	seventy two	setenta y dos
73	seventy three	setenta y tres
74	seventy four	setenta y cuatro
75	seventy five	setenta y cinco
76	seventy six	setenta y seis
77	seventy seven	setenta y siete
78	seventy eight	setenta y ocho
79	seventy nine	setenta y nueve
80	eighty	ochenta
81	eighty one	ochenta y uno

82	eighty two	ochenta y dos
83	eighty three	ochenta y tres
84	eighty four	ochenta y cuatro
85	eighty five	ochenta y cinco
86	eighty six	ochenta y seis
87	eight seven	ochenta y siete
88	eighty eight	ochenta y ocho
89	eighty nine	ochenta y nueve
90	ninety	noventa
91	ninety one	noventa y uno
92	ninety two	noventa y dos
93	ninety three	noventa y tres
94	ninety four	noventa y cuatro
95	ninety five	noventa y cinco
96	ninety six	noventa y seis
97	ninety seven	noventa y siete
98	ninety eight	noventa y ocho
99	ninety nine	noventa y nueve

**Letters (not yet recorded
in English.VAP or
Spanish.VAP)**

100	A
101	B
102	C
103	D
104	E
105	F
106	G
107	H
108	I
109	J
110	K
111	L
112	M
113	N
114	O
115	P
116	Q
117	R
118	S
119	T
120	U
121	V
122	W
123	X
124	Y
125	Z

Months

201	January
202	February
203	March

Meses

De Enero
De Febrero
De Marzo

204	April	De Abril
205	May	De Mayo
206	June	De Junio
207	July	De Julio
208	August	De Agosto
209	September	De Septiembre
210	October	De Octubre
211	November	De Noviembre
212	December	De Diciembre
	Days (ordinals)	Dias
301	First	El Primero
302	Second	El Segundo
303	Third	El Tercero
304	Fourth	El cuatro
305	Fifth	El cinco
306	Sixth	el Seis
307	Seventh	el Siete
308	Eighth	el ocho
309	Nineth	el Nueve
310	Tenth	el Diez
311	Eleventh	el Once
312	Twelvth	el Doce
313	Thirteenth	el Trece
314	Fourteenth	el Catorce
315	Fifteenth	el Quince
316	Sixteenth	el dieciséis
317	Seventeenth	el Diecisiete
318	Eighteenth	el Dieciocho
319	Nineteenth	el Diecinueve
320	Twentieth	el veinte
321	Twenty First	el veintiuno
322	Twenty Second	el veintidós
323	Twenty Third	el veintitrés
324	Twenty fourth	el veinticuatro
325	Twenty fifthe	el veinticinco
326	Twenty sixth	el veintiséis
327	Twenty seventh	el veintisiete
328	Twenty eighth	el veintiocho
329	Twenty nineth	el veintinueve
330	Thirtieth	el treinta
331	Thirty first	el treinta y uno
	Note: 332-399 has not been recorded yet in English.VAP or Spanish.VAP. You will need to record these for the PlayOrdinal to work properly.	
...		
399	Ninety ninth	
	Days of Week	
350	Sunday	Domingo
351	Monday	Lunes
352	Tuesday	Martes

353	Wednesday	Miercoles
354	Thursday	Jueves
355	Friday	Viernes
356	Saturday	Sabado
	Spanish Only	Solamente Espanol
400	n/a	Cien
401	n/a	Ciento
402	n/a	Doscientos
403	n/a	Trescientos
404	n/a	Cuatrocientos
405	n/a	Quinientos
406	n/a	Seiscientos
407	n/a	Setecientos
408	n/a	Ochocientos
409	n/a	Novcientos
	Currency	Dinero
500	Dollars	Dolares
501	Dollars and	Dolares y
502	cents	Centavos
503	hundred	n/a
504	thousand	mil
505	million	n/a
507	dollar	Un Dolar
508	dollar and	Un Dolar y
509	cent	Un Centavo
510	zero dollars	n/a
	Time	Tiempo
511	AM.	am
512	PM.	pm
513	O'Clock.	n/a
514	...point...	Punto
521	OH One.	n/a
522	OH Two.	n/a
523	OH Three.	n/a
524	OH Four.	n/a
525	OH Five.	n/a
526	OH Six.	n/a
527	OH Seven.	n/a
528	OH Eight.	n/a
529	OH Nine.	n/a
	Spanish Only	Solamente Espanol
601	n/a	Horas
602	n/a	Minutos
603	n/a	una
604	n/a	un Minuto
605	n/a	Milliones
606	n/a	Un Million
607	n/a	De Dolares
608	n/a	De Dolares y

609	n/a	Mil novecientos
610	n/a	Dos mil
700	Ringing sound	
701	Busy sound	
702	Fast busy sound	
703	Comfort for CSP (VoiceReco)	
710	Dial out test	
	Make Recording Prompts	
800	Press 1 for play 2 for record	
801	Enter prompt number Please enter the system	
802	password followed by the # sign	
803	For English press 1, for Spanish press 2	

In order to get the VAP file tools, you can download it from the web site. The download includes ENGLISH.VAP, SPANISH.VAP and a free utility that we wrote called VUTIL. (c++ source code for VUTIL is included). VUTIL is a command line application to import and export WAV files into any VAP file.

Here is how VUTIL works. Open a command prompt. Simply type VUTIL and press enter. You will see the following help:

USAGE:

```
VUTIL -CREATE <index file> <max number of entries>
VUTIL -IMPORT <index file> <file to import> <prompt #> <comment>
VUTIL -EXPORT <index file> <file to export to> <prompt #>
VUTIL -COMMENT <index file> <prompt #> <comment>
VUTIL -COMPRESS <index file>
VUTIL -UPDATEDIR <index file> <prefix> <suffix>
VUTIL -EXPORTDIR <index file> <prefix> <suffix>
VUTIL -LIST <index file>
```

In all of the commands, <index file> refers to the VAP file name. (i.e. ENGLISH.VAP)

-CREATE

This will create a new empty VAP file for you to import prompts into. Enter the maximum number of indexes that you will use. For example if the highest number will be 2000 you could perform the following:

```
VUTIL -CREATE NEW.VAP 2000
```

You do not need to load all 2000 voice prompts. You can skip entries and not use them.

-IMPORT

This will load a single wave file into the VAP for a specific index. For example, if you wanted to load hello into prompt 1000:

```
VUTIL -IMPORT NEW.VAP hello.wav 1000 "Hello file"
```

-EXPORT

This will take a file that is already in a VAP file and export it to a WAVE file. For example to export the "zero" prompt from ENGLISH.VAP:

```
VUTIL -EXPORT ENGLISH.VAP 0.WAV 0
```

-COMMENT

Add a comment to an index. This puts the comment in the VAP file that you can then see when you do a -LIST

-UPDATEDIR

This allows you to bulk load a VAP file with all the WAV files that you have in a folder. The names of the WAV files must be the prompt number (index) that it should be loaded to. These files can have a prefix before the number. For example, lets say I wanted to bulk load prompt 1000 – 1005 into our new VAP file. You could have the following files in a folder:

```
E1000.WAV  
E1001.WAV  
E1002.WAV  
E1003.WAV  
E1004.WAV  
E1005.WAV
```

```
VUTIL -UPDATEDIR NEW.VAP E WAV
```

This will load any file starting with an E and ending in a WAV with a number in between and load it into that position in the VAP file, replacing and segment already there.

-EXPORTDIR

This will take all the files in the VAP and export them into individual WAV files. You can specify the prefix and extension. For example:

```
VUTIL -EXPORTDIR ENGLISH.VAP E WAV
```

This will create Exxx.WAV for each segment loaded in the VAP, skipping any numbers where nothing was ever loaded.

-LIST

Creates a report of anything loaded in the VAP. For example:

```
VUTIL -LIST ENGLISH.VAP >LIST.TXT
```

This will list each prompt loaded in the VAP along with any Comments and redirects the output to the file LIST.TXT. Without the redirect file, the report would be displayed on the screen

Daily Statistics Report

The engine has the capability of e-mailing you a system usage report automatically at midnight. A sample stats report looks like this:

```
Report Run Date: 2005-06-28 00:00:05.000917 Daily Statistical Report
for: YOURBOX for 6/27/2005 Machine Up Time: 12 Days 5 Hours 6 Minutes
CTI32 Starts: 0
CTI32 Stops: 0
```

Incoming Calls:

```
Number of Calls: 929
Total Time      : 1028.8 (mins)
Avg Call Time   : 66 (secs)
```

Outgoing Calls:

```
Number of Calls: 300
Total Time      : 109.3 (mins)
Avg Call Time   : 21 (secs)
LD Calls        : 0
LD Time         : 0.0
```

```
Total Calls      : 1229
Total Time       : 1138.1 (mins)
```

```
Most Channels used at once: 16
```

Calls by Channel:

Chan	IN	Time	OUT	Time	Device / Last Activity
1	67	54.4	0	0.0	dtiB1T1 2005-06-27 23:00:09.000120
2	2	1.6	0	0.0	dtiB1T2 2005-06-27 17:00:14.000495
12	0	0.0	32	5.9	dtiB1T12 2005-06-27 23:00:07.000932
13	186	222.8	0	0.0	dtiB2T1 2005-06-27 23:31:43.000385
14	64	59.0	0	0.0	dtiB2T2 2005-06-27 23:08:36.000667
15	8	6.7	0	0.0	dtiB2T3 2005-06-27 23:00:09.000010
24	0	0.0	26	2.9	dtiB2T12 2005-06-27 23:00:08.000042
25	117	131.6	0	0.0	dtiB3T1 2005-06-27 23:52:20.000870
26	19	19.6	0	0.0	dtiB3T2 2005-06-27 23:17:40.000104
27	4	1.7	0	0.0	dtiB3T3 2005-06-27 19:00:07.000448
36	0	0.0	41	12.2	dtiB3T12 2005-06-27 23:00:06.000510
37	30	8.7	0	0.0	dtiB4T1 2005-06-27 23:00:06.000167
48	0	0.0	27	9.8	dtiB4T12 2005-06-27 23:00:05.000198
49	47	27.4	0	0.0	dtiB5T1 2005-06-27 23:45:54.000635

50	1	0.6	0	0.0	dtiB5T2	2005-06-27	15:08:06.000995
60	0	0.0	26	2.8	dtiB5T12	2005-06-27	23:00:05.000307
61	256	335.4	0	0.0	dtiB6T1	2005-06-27	23:00:08.000245
62	98	124.1	0	0.0	dtiB6T2	2005-06-27	22:38:13.000229
63	25	27.1	0	0.0	dtiB6T3	2005-06-27	22:08:35.000526
64	4	7.0	0	0.0	dtiB6T4	2005-06-27	20:06:56.000151
65	1	1.5	0	0.0	dtiB6T5	2005-06-27	17:54:45.000479
71	0	0.0	4	0.6	dtiB6T11	2005-06-27	16:33:22.000526
72	0	0.0	81	25.9	dtiB6T12	2005-06-27	23:00:07.000276
94	0	0.0	1	0.7	dtiB7T22	2005-06-27	16:37:12.000885
95	0	0.0	62	48.6	dtiB7T23	2005-06-27	23:31:12.000964

10 Most used Incoming DNIS:

5025555555	154
5025554000	117
5025550303	69
8595552000	55
5025550302	44
5025556000	42
2705556000	38
5025550301	37
8595556000	36
6155555555	31

Errors/Warnings Detected:

Qty	Code	Description
15	9780	dtiB2T1: could not get ANI - DNIS:5028490308
1	1895	dtiB6T1: DETECTED DISCONNECT DURING CALL SETUP

To Enable these statistics, use the Configuration Utility. Make sure you have the following set correctly:

- Set SendStatsReport to true
- Set EmailServer to a valid SMTP server that will allow you to relay e-mail
- Set EmailTo to the e-mail addresses that you want to receive the above report. You can comma separate the entries (i.e. me@ourname.com, you@ourname.com)
- Set EmailFrom to the e-mail address that should show in the From: field of the email.

Note: If there is no activity for the day, an e-mail will not be sent.

If you are not using the CTI32 Engine, you can use the .net or DLL commands to produce the report yourself. Please contact technical support. They can send you some sample code or the code to the CTI32Engine.

Use CTISetStatistic once to enable the stats. Call CTISetStatistic once for each channel. Call CTISetStatistic each time you get a DNIS or ANI. Then once per day, you run CTISendStatsReport which will convert the STATS file into a RPT text file. Also refer to the CTI32.DLL.PDF documentation for more information.

Automatically starting on bootup

Once you have your application debugged and in production, you will desire to have the system automatically start when you reboot the system. There are several issues with this. Since CTI32 relies on the Intel / Dialogic driver running before executing, you would not want to start opening ports until this service is started and ready. Also, your application may depend on other resources such as a database server to be started.

We have you covered. Here are the steps you must take to have everything start automatically:

- In the Windows “Services” utility, make sure the Startup type is set to automatic. (Start / Control Panel / Administrative Tools / Services) Find the “CTI32 Telephony Engine”, right click – select Properties.
- In the Configuration Utility (CTI32Config) set PauseBeforeExecution to how many seconds that you want the Engine to wait for the database engine to start. (or other application dependency) Don’t worry about the Intel / Dialogic driver as the Engine will wait for it to start when it calls cti.Start.
- You can also add a new multi string value (DependOnService) to the registry key called, HKLM\SYSTEM\CurrentControlSet\Services\Cti32svc. Here you can put the names of the services that must be started before the Cti32Svc can start. The OS will then take care of things automatically. For example: To ensure that the dialogic system is started before CTI, create the multi string value “DependOnService” and add the string “Dialogic” to the list. To add SQL server, add the string MSSQLSERVER. (Note that the list must have the names of the services in the registry list, not their display names.) To ensure you did everything correctly, go to the services panel and select the properties of CTI32 and select the Dependency Tab. You should see the services that must be started before CTI will start. WARNING: Modify the registry at your own risk.

Enabling Analog CallerID

Getting CTI32 to receive the CallerID on an analog involves the following steps:

- In the Configuration Utility, set LookForAnalogCallerID to true
- Set PerfectCall to 2 or 3 (Add up all the features you want enabled if you are using CSP)

This will automatically cause the engine to answer on the second ring – as the CallerID comes in between the first and second ring.

The entire caller ID string (name and number) will be found in lineData.ani.

If you are not using the engine, please call technical support for a copy of the CTI32Engine code so that you can follow the steps to enabling Analog caller ID. Also refer to the CTI32.DLL.PDF documentation for more information.

Dialing Masks

The .NET version of CTI32 supports Dialing Rules to help you with the endless anomalies of how to dial a phone number. Sometimes we dial 7-digits for a local phone number and sometime we dial a 10-digit phone number. Sometimes we need a 1 in front of the number and sometimes we don't. Sometimes dialing into one area code, certain prefix numbers are local while others are long distance. You may have some ports hooked up to one phone company and other ports with another and the rules are different between the two. The dialing masks are here to solve all of these problems.

In the Configuration Utility:

- Set the NumDialMasks in the default section. Each port that uses a dialing mask will refer to the mask name (ID). Each Dialing Mask consists of one or more rules. The NumDialMasks refers to the number of unique mask names (ID's) defined in the cti32engine.config XML file.
- Set each port's "UseDialMask" to the mask name (ID) it is to use. (i.e. Mask1). The easiest way to do this is to define it as you use the "Reconfigure Boards Section". Enter the Dialing Rule Name in the top blank of Panel 2.
- Click the Reconfigure Dial Masks and set up the Number of Rules and specify each rule. (see "how it works" below)

HOW IT WORKS

Here is how the computer applies the defined dialing rules:

- 1) The dialing **mask** is located and the number of rules for that **mask** is determined.
- 2) The logic rolls through each rule starting from the first rule
- 3) The source and destination is loaded.
- 4) If the source is empty, that means match that rule no matter what.
- 5) If the phone number starts with the digits in the source, then the digits in the destination are replaced with those digits. A destination of empty means to eliminate the matched characters.

In the above example, if you use always use a 10-digit phone, any calls to area code 303 will only use the last 7 digits. All other area codes would be preceded by a 1.

Talking to the Server from your own Windows or WEB application

Since the CTI32 Engine runs as a "service", it is not interactive. There are many instances where you may want to display information or statistics on what is happening as calls are going out or coming in. Perhaps you want to view sales per day, or calls made so far today. Perhaps you want to have the user kick off a custom process in your dispatch thread by pressing a button. For example, you could have a user import phone numbers into your database and then press a button to start a job to dial those phone numbers.

All of these activities require a user interface that you may wish to build. We have written a class that makes it easy to communicate with the CTI32 Engine. That class is

called CTI32StatusClient. It uses sockets to communicate with the server. Therefore, your application can either run on the same machine as the Engine or on a different machine.

The TMonitor program was written using the CTI32StatusClient class.

A demo is under development that will demonstrate how to do this. Basically, here are the steps to success:

- Instantiate the CtiConnection class by passing in the server IP, UserName, Password, Port to talk on, and an optional log file. (i.e. `ctiConnection=new CtiConnection(Location,UserName,Password, Port, "TMonitorClient.log");`)
- Set a callback (delegate) in your code that the class will call each time a name/value pair changes. There is a callback for a global name/value pairs and for each port name/value pairs that change. You may recall that your CTI32 application can set any name/value pair at any time.

Example for setting the callbacks and starting:

```
CtiConnection.PostGlobalNameChange gblNameChange= new  
CtiConnection.PostGlobalNameChange (this.GlobalNameChange);
```

```
CtiConnection.PostPortNameChange portNameChange = new  
CtiConnection.PostPortNameChange (this.PortNameChange);
```

```
CtiConnection.CtiConnectionStart (5, gblNameChange, portNameChange);
```

You can set the interval on how often the client polls the server for updates. Initially we had the server send data down each time a value changed, but we found this to be too resource intensive.

To stop the connection, you can issue the following command:

```
ctiConnection.CtiConnectionStop();
```

You can change the poll interval as follows:

```
ctiConnection.ChangeInterval (newinterval);
```

Sending a Command to the Server

You can send a custom command to the server as follows:

```
ctiConnection.UserCommand("Parm1","Parm2","parm3","parm4","parm5");
```

You have 5 variables that you can send with your user command. When the server receives this command it puts the information into the dispatchUserQueue using an Enqueue passing a structure with the parameters that you have sent. The Queue must have been created by your dispatch thread and placed in `cfg.dispatchUserQueue`. Then it is up to your dispatch thread to pull from that queue and execute your custom command.

If you need to communicate back to the client application, you can do that by having the dispatch queue set a name / value pair. The client application can look for the name / value pair and test the value for the response.

Speech Recognition

CTI32 has limited support for Speech Recognition. Here are the requirements:

- You must have a board (or HMP) that supports CSP (Intel / Dialogic's Continuous Speech Processing) This feature allows you to simultaneously play a file and record. It also allows for barge-in. Barge-in is forcing the play to end when the speaker says something.
- You must have a Speech Recognition engine. So far, we have only tested with the LumenVox SRE (www.lumenvox.com)

The CSP capability on the board only allows for capturing the speech stream. It does not actually recognize the pattern. For this you need a third party speech recognition engine like LumenVox. If you want to use a different recognition engine, please contact us.

Currently CTI32 has one method the supports speech recognition. It is the PlayAndRecognizeWord function. With this function, you pass in the GrammerFile of words to recognize for that particular menu, along with the standard play file information. The function returns the recognized word and a score from 1 to 1000 of how closely of a match it was to the word.

Note: CTI32 does not currently support recognition of a continuous stream of numbers.

There is one parameter that needs further explanation: EnableBargeIn. Since Speech Recognition packages are expensive and they usually charge by the concurrent connection, we have worked very hard to allow for multiplexing to those resources. For example, if you had a basic 24 port system, 24 ports of concurrent recognition resources would cost about \$12,000. Since not all ports are performing recognitions at the exact same time, perhaps your application could get along with 4 ports which may only cost about \$2000. Here is how you control this feature:

- Passing a 0 in EnableBargeIn will open a stream to the recognition engine as soon as the prompt begins playing. You will need close to a one for one ratio of recognition resources to voice resources. This option disables the Dialogic barge-in and uses the barge-in of the recognition engine.
- Passing a 1 in EnableBargeIn will enable the dialogic barge-in. It will not use a recognition resource until the Dialogic CSP resource detects a user utterance (otherwise known as barge-in)
- Passing a 2 in EnableBargeIn will act just like option 1, but will not allow the barge-in until after the play prompt has completed playing the whole entire message.

One important note: If you are using PRI lines, Dialogic has a huge limitation on some of their boards (like the JCT board). The PRI firmware load is not compatible with the

CSP firmware load. Therefore a single virtual board cannot both load CSP and PRI on the voice resources. You would therefore have to load PRI on one board and CSP on another board. The voice resource that go with the PRI channels will still work for PRI but only for play, record, get digits etc. They will not perform simultaneous Play/record – thus these expensive resources become useless for speech recognition applications. Therefore many people purchase D/480JCT-T1 boards and put the CSP on the one virtual board and the PRI on the other virtual board. You gotta love Dialogic. Supposedly, if you purchase the DM/V (DM3) boards, this is not a problem.

So here is how you set up a CSP / LumenVox application.

- Make sure you have a CSP compatible board
- In the PerfectCall parameter (add 4 or 8 to the existing value) 4 is for a springware board (i.e. JCT board) and 8 is for a DM/V board
- Edit the cti32.ini file. Make sure you have a the following sections. Set each as appropriate to your system.

```
[VoiceRecognition]
EngineDll = ..\\CTILumenVox\\Debug\\CTILumenVox.dll
```

```
[LumenVox]
NumberOfPorts=2
IPAddress=172.18.13.200
```

```
[CSP]
DXCH_EC_TAP_LENGTH=128
ECCH_NLP=1
DXCH_BARGEINONLY=0
ECCH_SVAD=1
DXCH_SPEECHPLAYTHRESH=-40
DXCH_SPEECHPLAYTRIGG=6
ECCH_XFERBUFFERSIZE=5120
```

This has been set up so that in the future, we can write interfaces to other recognition engines besides LumenVox. The EngineDll will be replaced with another interface.

Playing Files from a Network Drive or Mapped Drive

The CTI32Engine runs as a service. As such it runs under its own credentials and process attributes. By default the CTI32Engine is installed to run under the Local System account. The Local System Account does not have access to the network. To allow the service to have access, you need to go to the Windows Services SnapIn and change the properties of the CTI32 Service to run under credentials that have Network Access.

This will allow you to play files on the network using a valid (and accessible) network path such as: <\\myremote\playfiles\abc.wav>

Every process runs within its own environment. Since the account specified in the services SnapIn doesn't actually go through the "login process", mapped drives created interactively under the same account are not restored to the service's environment even if you told them to restore automatically. Accordingly, you must map the drive in your code if you plan to use mapped drives. CTI has a class for doing this: MapNetworkDrive (Available with Version 4.09). The class exposes Map and UnMap methods. To map a drive simply use the following code snippet:

```
try
{
    MapNetworkDrive.Map("M:", @"\\myremote\playfiles");
}
catch (Exception)
{
    log.Write("Error Mapping Drive");
}
```

To unmap:

```
MapNetworkDrive.UnMap("M:");
```

After the map is made, you can refer to the files via the mapped path. I.E.: M:\abc.wav

By default, the LOCAL Administrator account won't work in a DOMAIN environment; you'd need to use the DOMAIN Administrator account or another account with network privileges to the other machine. Likewise, the account needs privileges to run as a service on the local machine.

If you are using a WORKGROUP you can use the local Administrator account, but both machines HAVE to have the same Password, and the sharing needs to be setup properly on the target machine.

Inventive Labs does not recommend using an Administrator account in a production environment. As always, you should take into account the security environment you work in and select an appropriate account for the service and its network access needs.

CTI32.INI settings

General ini Settings

FailOnProgressingCauseValue

This parameter is no longer being used

SkipIsdnRestart

When the system shutdowns (ISDN-PRI only), it issues a dialogic api call `cc_Restart`. Also, when the system starts and `SetChannelState` is true, it issues this command. If this value is set to 1, then each `cc_Restart` api call is skipped on both shutdown and startup.

If you are having problems with the lines not answering after you stop/start the CTI32 Service. Set this parameter to 1.

SkipIsdnOutOfService

On shutdown, or restart, the ISDN-PRI channel state is set to `OUT_OF_SERVICE`. Setting this to 1, prevents that from happening.

If you are having problems with the lines not answering after you stop/start the CTI32 Service. Set this parameter to 1.

BbitSignaling

(For T1 installations) This is an option set for a specific switch (a InterTel switch) which watches the B-bit transitions as well as the A bit transitions. If you wish to get events on B-bit transitions as well as A-bit transitions, set this parameter to 1.

GetDigitsLcoffOption

This setting specifies how `GetDigits` behaves. Allowable settings are:

- 0 Legacy Mode: `TM_LCOFF` is returned immediately (and no digits are returned) if the line is hung-up.
- 1 Delayed Mode: `TM_LCOFF` is returned but the system still attempts to get the digits from the buffer and return them to the user.
- 2 Off Mode: `TM_LCOFF` is never returned from a call to `GetDigits`.

SkipDriverLoadedCheck

This setting specifies if the engine will skip the check to see if dialogic is running and loaded. Allowable settings are:

- 0 Always checks to see if the driver is loaded.
- 1 Skips check.

LogEventStats

This setting specifies if the engine will log statistics about how the events thread is performing. Allowable settings are:

- 0 No Stats are logged.
- 1 Logs Stats.

EventsThreadPriority

This setting specifies if the engine will raise the priority for the events thread. Allowable settings are:

- 0 No boost in priority. (Default)
- 1 Small boost in priority.
- 2 Large boost in priority.

GCDropCallWaitTime

This setting specifies the length of time the system will wait for the GCEV_DROP_CALL event to complete. Allowable settings are from 5 to 60 seconds.

CCDropCallWaitTime

This setting specifies the length of time the system will wait for the CCEV_DROP_CALL event to complete. Allowable settings are from 5 to 60 seconds.

AllowProceedingToStartCPA

This setting allows the engine to start call progress analysis at the occurrence of the PROCEEDING event. This is very helpful with cell phone calls that often go from PROCEEDING to CONNECTED skipping the ALERTING phase. The default is to start CPA on the ALERTING or CONNECTED event, whichever comes first. Allowable settings are:

- 0 Default. CPA starts at ALERTING or CONNECT.
- 1 CPA start at PROCEEDING, ALERTING or CONNECT.

Voice Recognition ini Settings

EngineDll

This setting tells CTI32 what dll to use for voice recognition functions.

Example:

```
EngineDll=..\CTILumenVox\Debug\CTILumenVox.dll
```

LumenVox ini Settings

NumberOfPorts

If using the LumenVox Voice Recognition Library, use this setting to tell CTI32 how many ports you have licensed from LumenVox.

IPAddress

If using the LumenVox Voice Recognition Library, use this setting to tell CTI32 what the IP Address is for the the LumenVox licensing server.

LicenseType

This can be SpeechPort for a full license or VoxLite for the Lite version of LumenVox.

CSP ini Settings

DXCH_EC_TAP_LENGTH
ECCH_NLP
DXCH_BARGEINONLY
ECCH_SVAD
DXCH_SPEECHPLAYTHRESH
DXCH_SPEECHPLAYTRIGG
ECCH_XFERBUFFERSIZE

Please refer to the Continuous Speech Processing API Programming Guide at http://resource.intel.com/telecom/support/releases/winnt/sr60pci/onldoc/htmlfiles/csp_programming.html for more information about these settings.

Vendetta ini Settings

DTMF_FWD_TWIST
DTMF_REV_TWIST
DTMF_2ND_HARM_RATIO_LO
DTMF_2ND_HARM_RATIO_HI
DTMF_TONE_TO_TOTAL_C1
DTMF_TONE_TO_TOTAL_C2
DTMF_TONE_TO_TOTAL_C3
DTMF_TOWER_LO
DTMF_TOWER_HI
DTMF_CLAMP_VOLUME
DTMF_DELAY

Please refer to your Vendetta configuration documentation for more information about these settings.

HMP ini Settings

Realm, Identity, Username and Password

These settings are used for MD5 authentication in situations where it is necessary to provide credentials. Typically, you will need these settings when using a proxy server or other type of softswitch that challenges the system during the registration process or while making an outbound call.

Example:

```
Realm=inventivelabs.com
Identity=sip:hmpuser@sip.inventivelabs.net
Username=hmpuser
Password=iAbgM57M
```

Notes:

These parameters are required when the SipRegister setting is used.

RegisterLocalAlias

This setting specifies the alias for the local machine, as required in the registration process.

Example:

```
RegisterLocalAlias=sip:hmpuser@12.151.50.94:5060
```

Notes:

This parameter is required when the SipRegister setting is used.

RegisterServer

This is the proxy server or softswitch at which you wish to register. This address or URL is provided to you by your service provider.

Example:

```
RegisterServer=sip:sip.inventivelabs.com
```

Notes:

This parameter is required when the SipRegister setting is used.

RegisterClient

This is the name by which you are known to the proxy server or softswitch.

Example:

```
RegisterClient=sip:hmpuser@sip.inventivelabs.com
```

Notes:

This parameter is required when the SipRegister setting is used.

RegisterOutboundProxyHost, RegisterOutboundProxyPort

This is the proxy that is used for all outbound calls. If this is blank, no proxy is used and the calls originate from the CTI server.

Example:

```
RegisterOutboundProxyHost=sip.inventivelabs.com  
RegisterOutboundProxyPort=5060
```

Notes:

RegisterOutboundProxyHost may still be required even if you do use the SipRegister setting. This setting can be blank.

InviteFromHeader

If your proxy server requires a "From" Header, you can use this setting to provide one. This is used for placing outbound calls.

Example:

```
InviteFromHeader=From: Hmp <sip:hmpuser@sip.inventivelabs.com>
```

Notes:

InviteFromHeader may still be required even if you do not use the SipRegister setting. This setting can be blank.

InviteToSuffix

When making outbound calls, the INVITE request includes both the number you wish to dial and the proxy you are using to place the call through. To specify the proxy, use this setting.

Example:

```
InviteToSuffix=@sip.jnctn.net
```

Notes:

This setting can be blank.

SipRegister

This setting forces the HMP system to register with a proxy server. Allowable settings are 1 and 0.

Example:

SipRegister=1

Notes:

If this setting is 1 then Realm, Identity, Username, Password, RegisterLocalAlias, RegisterServer, and RegisterClient are required.

SpoofName

This setting is used to specify the CallerID Name that is sent when calling other IP addresses. This has no effect on calls placed on the PSTN.

Example:

SpoofName=Intel HMP

Notes:

This setting can be blank.

DTMF_Type

This setting specifies how DTMF is handled by the HMP server. Allowable settings are:

0	Both In-band and Out-of-Band are supported.
1	Inband Only
2	Out-of-Band (RFC 2833) Only

Example:

DTMF_Type=1

Notes:

This setting is required.

InitializeForSip

When using HMP, this setting MUST be set to 1 to allow the library to initialize properly.

0	Not using HMP
1	Using HMP

Example:

InitializeForSip=1

Notes:

This setting is required.

CodecOrder

This setting allows you to specify what codecs to use as well as the preference given to the codecs. Those listed first are preferred to those listed later. Allowable codecs are:

```
g711Ulaw64k
g711Alaw64k
g7231_5_3k
g7231_6_3k
g729AnnexA
g729AnnexAwAnnexB
t38UDPFax
```

Example:

```
CodecOrder=g711Ulaw64k t38UDPFax
```

Notes:

This setting is required. If you are using fax, t38UDPFax is required.

3PCC

```
0 Initialize HMP in 1PCC (First Party Call Control Mode)
1 Inititalize HMP in 3PCC (Third Party Call Control Mode)
```

SipMaxCalls

This setting specifies the maximum number of simultaneous SIP calls that can occur on the system at one time. The default setting is 0, which has HMP autoconfigure up to 120 ports. If you have more than 120 ports, you must specify the correct value here. Also, if you are using the autoconfigure option in CTI32Engine, then you must specify the number of ports you have licenses for here even if you have less than 120 ports.

Contact

This setting allows you to explicitly specify the contact header that is transmitted with each SIP message. The default is to let HMP construct one it thinks is best.

Example:

```
Contact=Contact: <sip:abc123@99.111.206.98:5060>
```

SipSignalingIp

This allows configuration of the IP address to use for the RTP streams. (For use by systems with multiple NIC cards.) If left blank the system picks the NIC card for RTP traffic.

Example:

```
SipSignalingIp=12.151.50.50
```

SipSignalingPort

This allows configuration of the IP signaling port. If left blank the system uses 5060 by default.

Example:

```
SipSignalingPort=5060
```

HandleReqModifyCall

This allows the system to accept re-invites from other systems when in 1PCC (First Party Call Control). It also allows for automatically switching to T.38 when requested by remote systems. Allowable values are 0 and 1.

Example:

```
HandleReqModifyCall=1
```

FAX ini Settings

FAX_API

This setting specifies which fax libraries CTI32 uses for faxing capabilities. Allowable settings are:

- 0 Gamma Fax Libraries
- 1 Intel R4 Fax Libraries
- 2 HMP t38 Fax Libraries
- 3 HMP/Intel R4 Fax Manual Mode

Example:

```
FAX_API=2
```

Notes:

This setting is required if using fax.

FAX_HMPMODE

If using HMP t38, you must specify who will issue the re-INVITE to switch codecs. Allowable settings are:

0	Slave
1	Master

Example:

```
FAX_HMPMODE=1
```

Notes:

Switching codecs occurs during the CTI.Route command. If you are 'slave' then the route command will wait up to 60 seconds for the codec to change. If you are the master, then the re-INVITE will be issued as soon as the CTI.Route command is issued. The system will wait 60 seconds for the remote site to respond to the re-INVITE.

FAX_USEEXPLICITOPEN

This setting applies to FAX_API settings of 1 and 2. By default the system only looks for fax resources on the voice resources that are opened by the user. In certain situations, such as DM3 boards, where the fax resources are separate from the voice resources, you can explicitly specify the fax boards to search for fax resources.

Allowable settings are:

0	(Default) System finds fax resources by searching the voice resources that are opened prior to calling CTIInitializeFax.
1	The system searches the boards specified in the FAX_EXPLICITBOARDS setting for fax resources.

Example

```
FAX_USEEXPLICITOPEN=1
```

Notes:

This setting is used in conjunction with the FAX_EXPLICITBOARDS string.

FAX_EXPLICITBOARDS

This setting applies to FAX_API settings of 1 and 2. The FAX_USEEXPLICITOPEN must be set to 1 for this setting to have an effect. This setting allows you to specify the specific boards you want the system to search for fax resources. Only the boards you specify will be searched. Separate boards by spaces.

Example

```
FAX_EXPLICITBOARDS=dxxxB7 dxxxB8
```

Notes:

Fax resources are often grouped in sets of 4 per virtual board. On certain DM3 boards, the fax resources follow the voice resources in numbering sequence. Therefore, if you know that the last voice resource is dxxxB6C4 and your board supports 24 fax resources, your fax resources are likely to be on virtual boards dxxxB7 thru dxxxB12.

TTS ini Settings

Name

Use this setting to specify the Name of the voice you wish to use for text to speech.

Example:

```
Name=Microsoft Mary
```

InitializeOnStart

Use this setting to specify the Name of the voice you wish to use for text to speech.

- 0 Do not Initialize TTS Engine At Startup, but initialize if a TTS function is called.
- 1 Initialize On Startup

VoiceFormat

Sets the voice format of the wave files produced by the TTS engine.

Valid Values are:

Kbps24	0	ADCPM 6Khz (VOX) 3000 bytes/sec
Kbps32	1	ADCPM 8Khz (VOX) 4000 bytes/sec
Kbps64	2	8Khz DecTalk Wav format 8000 b/s
WAV	3	
Wave6	4	6Khz Wave 6000 bytes/sec
Wave8	5	8Khz Wave 8000 bytes/sec
Wave11	6	11Khz Wave 11000 bytes/sec
G726	7	G.726 – 4000 bytes/sec
GSM610	8	GSM – about 1600+ bytes/sec
GSM11	9	
Wave8Mulaw	10	Wave 8Khz MuLaw
Wave11Mulaw	11	11Khz MuLaw
Wave8Alaw	12	Wave 8Khz ALaw
Wave11Alaw	13	Wave 11Khz ALaw
Vox8Mulaw	14	

Wave8X16	15	
TrueSpeech	16	TrueSpeech/Lucent CELP

Example:

```
VoiceFormat=11
```

ISDN SETTINGS in the CTI32.INI file

```
[ISDN]
;All values should be in HEX in the format 0x00 (except for LoadOptions
and sub phone numbers)
```

LoadOptions

```
0=Do not use values from INI file
1=Load once on start use values from INI file
2=Load Each Phone Call
```

InitializeBlock

The ISDN dialing procedure uses a MAKECALL block. The parameter (in Hex) specifies how to initialize each byte in the call block. Leave empty to not initialize.

BC_xfer_cap

```
#define BEAR_CAP_SPEECH           0x00  /* Speech Bearer Capability */
#define BEAR_CAP_UNREST_DIG       0x08  /* Unrestr Digital Capability */
#define BEAR_CAP_REST_DIG         0x09  /* Restr Digital Capability */
#define BEAR_CAP_3DOT1K_AUDIO     0x10  /* 3.1KHz Audio Capability */
#define BEAR_CAP_7K_AUDIO         0x11  /* 7KHz Audio Capability */
#define BEAR_CAP_VIDEO            0x18  /* Video Capability */
#define PACKET_TRANSFER_MODE      0x00
```

BC_xfer_mode

```
#define ISDN_ITM_CIRCUIT          0x00  /* info transfer mode - circuit
mode */
#define ISDN_ITM_PACKET          0x02  /* info transfer mode - packet
mode */
```

BC_xfer_rate

```
/*
 * Information Transfer Rate
 */
#define BEAR_RATE_64KBPS          0x10  /* B_CHANNEL_UNITS 1X64 */
#define BEAR_RATE_128KBPS        0x11  /* Non-standard 2X64 */
#define BEAR_RATE_384KBPS        0x13  /* H0_CHANNEL_UNITS 6X64 */
#define BEAR_RATE_1536KBPS       0x15  /* H11_CHANNEL_UNITS 24X64 */
#define BEAR_RATE_1920KBPS       0x17  /* H12_CHANNEL_UNITS 30X64 */
```

usrinfo_layer1_protocol

```
/*
 *   Bearer Capability Element
 */
#define ISDN_UI11_CCITTV110      0x01 /* user info layer 1 - CCITT
V.110/X.3
*/
#define ISDN_UI11_G711ULAW      0x02 /* user info layer 1 - G.711 u-
law */
#define ISDN_UI11_G711ALAW      0x03 /* user info layer 1 - G.711 A-
law */
#define ISDN_UI11_G721ADCPM      0x04 /* user info layer 1 - G.721
ADCPM */
#define ISDN_UI11_G722G725      0x05 /* user info layer 1 - G.722 and
G.725 */
#define ISDN_UI11_G722F725      0x05 /* user info layer 1 - G.722 and
G.725
*/
#define ISDN_UI11_H261          0x06 /* user info layer 1 - H.261 */
#define ISDN_UI11_NONCCITT      0x07 /* user info layer 1 - non-CCITT
*/
#define ISDN_UI11_CCITTV120      0x08 /* user info layer 1 - CCITT
V.120 */
#define ISDN_UI11_CCITTX31      0x09 /* user info layer 1 - CCITT
X.31 */
#define ISDN_UI11_DEFAULT      0xFF /* user info layer 1 - Default
Value
```

usr_rate

```
#define ISDN_UR_EINI460          0x00 /* user rate - E bits in I.460
*/
#define ISDN_UR_600              0x01 /* user rate - 0.6 kbits */
#define ISDN_UR_1200             0x02 /* user rate - 1.2 kbits */
#define ISDN_UR_2400             0x03 /* user rate - 2.4 kbits */
#define ISDN_UR_3600             0x04 /* user rate - 3.6 kbits */
#define ISDN_UR_4800             0x05 /* user rate - 4.8 kbits */
#define ISDN_UR_7200             0x06 /* user rate - 7.2 kbits */
#define ISDN_UR_8000             0x07 /* user rate - 8.0 kbits */
#define ISDN_UR_9600             0x08 /* user rate - 9.6 kbits */
#define ISDN_UR_14400            0x09 /* user rate - 14.4 kbits */
#define ISDN_UR_16000            0x0A /* user rate - 16.0 kbits */
#define ISDN_UR_19200            0x0B /* user rate - 19.2 kbits */
#define ISDN_UR_32000            0x0C /* user rate - 32 kbits */
#define ISDN_UR_48000            0x0E /* user rate - 48 kbits */
#define ISDN_UR_56000            0x0F /* user rate - 56 kbits */
#define ISDN_UR_64000            0x10 /* user rate - 64 kbits */
#define ISDN_UR_134              0x15 /* user rate - .1345 kbits */
#define ISDN_UR_100              0x16 /* user rate - .100 kbits */
#define ISDN_UR_75_1200          0x17 /* user rate - .075/1200 kbits
*/
#define ISDN_UR_1200_75          0x18 /* user rate - 1200/.075 kbits
*/
#define ISDN_UR_50               0x19 /* user rate - .050 kbits */
#define ISDN_UR_75               0x1A /* user rate - .075 kbits */
#define ISDN_UR_110             0x1B /* user rate - .110 kbits */
```

```

#define ISDN_UR_150           0x1C  /* user rate - .150 kbits */
#define ISDN_UR_200           0x1D  /* user rate - .200 kbits */
#define ISDN_UR_300           0x1E  /* user rate - .300 kbits */
#define ISDN_UR_12000         0x1F  /* user rate - 12 kbits */
#define ISDN_UR_DEFAULT      0xFF  /* user rate - Default */

destination_number_type

/*
 * Number types
 */
#define EN_BLOC_NUMBER        0x00  /* Number is sent en-bloc */
#define INTL_NUMBER           0x01  /* International number */
#define NAT_NUMBER            0x02  /* National number */
#define LOC_NUMBER            0x04  /* Local (directory) number */
#define OVERLAP_NUMBER        0x05  /* Number is sent overlap */

destination_number_plan

/*
 * Numbering plans
 */
#define UNKNOWN_NUMB_PLAN     0x00  /* Unknown plan */
#define ISDN_NUMB_PLAN        0x01  /* ISDN numb. plan E.164 */
#define TELEPHONY_NUMB_PLAN   0x02  /* Telephony numb. plan E.163 */
#define PRIVATE_NUMB_PLAN     0x09  /* Private numbering plan */

destination_sub_number_type

/*
 * CALLER_SUB_ADDR_IE and CALLED_SUB_ADDR_IE
 */
#define OSI_SUB_ADDR           0x00  /* OSI Sub-address */
#define USER_SPECIFIC_SUB_ADDR 0x02  /* User-specific Sub-address */
#define IA_5_FORMAT            0x50  /* IA 5 sub-address digit format
*/

destination_sub_phone_number

Usually leave this empty

destination_sub_number_plan

Usually set this to 0xFF

origination_number_type

/*
 * Number types
 */
#define EN_BLOC_NUMBER        0x00  /* Number is sent en-bloc */
#define INTL_NUMBER           0x01  /* International number */
#define NAT_NUMBER            0x02  /* National number */
#define LOC_NUMBER            0x04  /* Local (directory) number */
#define OVERLAP_NUMBER        0x05  /* Number is sent overlap */

```

origination_number_plan

```
/*
 * Numbering plans
 */
#define UNKNOWN_NUMB_PLAN      0x00 /* Unknown plan */
#define ISDN_NUMB_PLAN         0x01 /* ISDN numb. plan E.164 */
#define TELEPHONY_NUMB_PLAN    0x02 /* Telephony numb. plan E.163 */
#define PRIVATE_NUMB_PLAN      0x09 /* Private numbering plan */
```

origination_sub_number_type

```
/*
 * CALLER_SUB_ADDR_IE and CALLED_SUB_ADDR_IE
 */
#define OSI_SUB_ADDR           0x00 /* OSI Sub-address */
#define USER_SPECIFIC_SUB_ADDR 0x02 /* User-specific Sub-address */
#define IA_5_FORMAT            0x50 /* IA 5 sub-address digit format
 */
```

origination_sub_number_plan

```
/*
 * Numbering plans
 */
#define UNKNOWN_NUMB_PLAN      0x00 /* Unknown plan */
#define ISDN_NUMB_PLAN         0x01 /* ISDN numb. plan E.164 */
#define TELEPHONY_NUMB_PLAN    0x02 /* Telephony numb. plan E.163 */
#define PRIVATE_NUMB_PLAN      0x09 /* Private numbering plan */
```

origination_sub_phone_number

Usually leave empty

facility_feature_service

```
#define ISDN_FEATURE           0x00 /* feature/service - feature */
#define ISDN_SERVICE           0x01 /* feature/service - service */
#define ISDN_CPN_PREF          0x01 /* facility coding - CPN
preferred */
#define ISDN_BN_PREF           0x02 /* facility coding - BN
preferred */
#define ISDN_CPN               0x03 /* facility coding - CPN */
#define ISDN_BN                0x04 /* facility coding - BN */
```

facility_coding_value

```
#define ISDN_SDN               0x01 /* service coding - SDN */
#define ISDN_MEGACOM800        0x02 /* service coding - MEGACOM 800
 */
#define ISDN_MEGACOM           0x03 /* service coding - MEGACOM */
#define ISDN_WATS              0x04 /* service coding - WATS */
#define ISDN_TIE               0x05 /* service coding - TIE */
```

```

#define ISDN_ACCUNET          0x06  /* service coding - ACCUNET SDS
*/
#define ISDN_LONG_DS        0x07  /* service coding - Long
distance */
#define ISDN_INT_800        0x08  /* service coding - internation
800 */
#define ISDN_CA_TSC         0x09  /* service coding - CA TSC */
#define ISDN_ATT_MULTIQ     0x10  /* service coding - ATT
MultiQuest */
#define ISDN_TNID_USER      0x00  /* network id type - user
specified */
#define ISDN_TNID_NAT       0x02  /* network id type - national */
#define ISDN_TNID_INTER     0x03  /* network id type -
international */
#define ISDN_NIDPLN_UNK     0x00  /* network id plan - unknown */
#define ISDN_NIDPLN_CIC     0x01  /* network id plan - carrier id
code */
#define ISDN_NIDPLN_DNIC    0x03  /* network id plan - datanetwork
id co */

```

Defaults

There is no clear authority on what the default values of the call block are.

One document says this:

Dialogic contains an ISDN MAKECALL block which contains the data inserted into the ISDN SETUP message during an outbound call. The MAKECALL block pertinent parameters are shown below:

Data Type MAKECALL_BLK Structure Parameter Description Default Value

```

BYTE BC_xfer_cap Bearer Capability Info Transfer Capability (*) 0x00
BYTE BC_xfer_mode Bearer Capability Transfer Mode (*) 0x00
BYTE BC_xfer_rate Bearer Capability Transfer Rate (*) 0x10
BYTE usrinfo_layer1_protocol User Info Layer 1 Protocol (*) 0x00
BYTE usr_rate User Rate
BYTE destination_number_type Destination Number Type (*) 0x01
BYTE destination_number_plan Destination Number Plan (*) 0x01
BYTE destination_sub_number_type Destination Sub-Number Type 0xFF

char destination_sub_phone_number[MAXPHONENUM] Destination Sub-Number Plan
0xFF
BYTE origination_number_type Origination Number Type 0xFF
BYTE origination_number_plan Origination Number Plan 0xFF
BYTE origination_sub_number_type Origination Sub-Number Type 0xFF
char origination_sub_phone_number[MAXPHONENUM] Origination Sub-Number Plan
0xFF
BYTE facility_feature_service Facility Feature Service 0x00
BYTE facility_coding_value Facility Coding Value 0x00

```

It appears the ISDIAG utility defaults are set like this:

```
BC_xfer_cap=0x00
BC_xfer_mode=0x00
BC_xfer_rate=0x10
usrinfo_layer1_protocol=0x02
usr_rate=0xFF
destination_number_type=0x02
destination_number_plan=0x01
destination_sub_number_type=0xFF
destination_sub_phone_number=
destination_sub_number_plan=0xFF
origination_number_type=0xFF
origination_number_plan=0xFF
origination_sub_number_type=0xFF
origination_sub_number_plan=0xFF
origination_sub_phone_number=
facility_feature_service=0x01
facility_coding_value=0x03
```

Call Progress ini Settings

Attention! If you are using HMP or DM3 boards, you cannot use these values (Except SilenceEvents) to change CallProgress: See the "How to improve outbound Dialing Call Progress Results (HMP and DM/V boards)" in the next section of this guide.

Warning! If you enable custom values and then decide to disable, you need to restart the DCM so that the default values get reset.

Also, if you are using non-analog lines AND the CTI32Engine, you'll need to set the UseQualificationTemplates value in the CTI32Engine.config to 1 so that the voice resource boards are opened and configured. Call Progress Values for springware boards are set when the board is opened.

Enable

Use this setting to enable the call progress configuration of springware boards.

- 0 Do not enable the Springware configuration of Call Progress.
- 1 Enable the Springware configuration of Call Progress.

SilenceEvents

Use this setting in conjunction with CTIMonitorSilence to further enhance your call progress detection.

- 0 Do not enable the Springware configuration of Call Progress.
- 1 Enable the Springware configuration of Call Progress.

SilenceEvents=0
pvd_qual_qminsnr

For positive voice detect, the minimum allowable SNR for voice.

pvd_qual_qmaxsnr

For positive voice detect, the maximum allowable SNR for voice.

pvd_qual_maxpk

For positive voice detect, the maximum number of peaks for voice.

pvd_qual_maxring

For positive voice detect, the maximum number of frames for ringback not voice.

pvd_qual_ringthres

For positive voice detect, the signal to noise ratio for ringback.

pvd_qual_pvdwin

For positive voice detect, the number of frames in a window sample.

pvd_qual_pvdthresh

For positive voice detect, the minimum energy for voice.

pvd_qual_pvdrblow

For positive voice detect, the lower frequency of ringback.

pvd_qual_pvdrbhig

For positive voice detect, the upper frequency of ringback.

amd_qual_maxansiz

For answering machine detect, the size of answer #1.

amd_qual_maxans2

For answering machine detect, the size of answer #2.

amd_qual_maxans3

For answering machine detect, the size of answer #3.

amd_qual_lohiss

For answering machine detect, the low hiss (noise) range.

amd_qual_hihiss

For answering machine detect, the high hiss (noise).

amd_qual_bhparm

For answering machine detect, the noise below hiss ratio.

amd_qual_cvthr1

For answering machine detect, the cv threshold #1.

amd_qual_cvthr2

For answering machine detect, the cv threshold #2.

amd_qual_maxcvth

For answering machine detect, the maximum cv threshold.

amd_qual_nmaxbrod

For answering machine detect, the maximum broad band energy - noise.

amd_qual_nmaxerg

For answering machine detect, the size maximum total energy - noise.

amd_qual_maxsil

For answering machine detect, the maximum silence.

amd_qual_voice_thres

For answering machine detect, the voice threshold.

amd_qual_sil_thres

For answering machine detect, the silence threshold.

amd_qual_bandf_low

For answering machine detect, the frequency band filter, lower limit in hz.

amd_qual_bandf_high

For answering machine detect, the upper limit in hz.

Installing the Complimentary Dialogic Drivers (for Dialogic Springware boards)

Download from Intel the following:

<http://resource.intel.com/telecom/support/releases/winnt/index.html>

- Get the System Release 5.1.1 for Windows
- System Release 5.1.1 Service Pack 1

First install the System Release 5.1.1

- Execute setup
- Click Custom
- Click the following options:
 - Drivers, Firmware & Configuration Files
 - Performance Counters for Win NT Perf. Monitor
 - ISDN Package
 - GlobalCall API Package
- Select the following ISDN protocols
 - 4ESS
 - 5ESS
 - DMS
 - NI2

It will ask you to reboot. Go ahead and reboot.

Now install the Service Pack 1.

- Execute Setup

It will ask you to reboot. Go ahead and reboot.

After the system reboots, you can start the dialogic service using the DCM.

How to use Intel HMP and Xten Lite as a CTI32 Emulator

Sometimes you would like to perform your telephony development without having a Dialogic board. Perhaps you have a notebook computer and you would like to write your CTI32 applications while you are traveling. Using the Intel HMP software and the XTen softphone, this becomes possible. This document explains how to set this up to properly do this.

1. Download and Install the FREE Intel HMP Software. Get the latest version at this link:
<http://www.intel.com/design/network/products/telecom/software/index.htm#hmp>
Intel give you a complimentary license for a single port – perfect for our use as an emulator. Do not download the demo 4-port license, as this will expire after 30-days. The single port installation will not expire. The download is about 48MB. Select the “Core Runtime Package” as you are installing. (Requires reboot)
2. Run the Dialogic Configuration Manager (DCM) and start the service.
3. Download and install the HMP version of CTI32.
<http://www.inventivelabs.com/cti32/download.htm>
4. Run the CTI32 Configuration utility. Click on the “Reconfigure Boards” button along the top. Select 1 board. Select “Using HMP” and make sure it says 1 port at the bottom. Leave blank the Dialing Rule Name and T1/E1 options. Click “Finish Section”. Make sure the UseGlobalCall is set to “Yes”. Press the Save Button at the top of the screen. Click on the “Start Service” button on the bottom of the screen. After a few seconds, Click on the View CTI32Engine.log file – if everything is working right, you should see (1) Waiting for Call at the end of this file.
5. Download and Install the FREE X-Lite Softphone using this link:
<http://www.xten.com/index.php?menu=products&smenu=download> Click the menu icon and set the settings as described below.

Configuration Settings for XTen Lite (Version 2.0 build 1103m)

System Settings:

Network

Auto Detect IP: No
Listen on IP
Use X-Nat to choose SIP/RTP Ports: Never
Listen on SIP Port: 5080
Listen on RTP Port: 8000
NAT Firewall IP:
Outbound SIP Proxy
Force Firewall Type: Do Not Force
Primary STUN Server
Secondary STUN Server: xten.net

Primary DNS Server
Secondary DNS Server
Provider DNS Server: dns.xten.net

SIP Proxy

Default:

Enabled: Yes
Display Name: "UserName"
Username:
Authorization User
Password:
Domain/Realm:
SIP Proxy:
Outbound Proxy:
Use outbound Proxy: Never
Send Internal IP: Never
Register: Never
Voicemail SIP URL
Forward SIP URL
Use Voicemail: Forward to Voicemail
Direct Dial IP: Yes
Dial Prefix:
Provider website:
Update Settings:
Reset

X-Tunnels Host:
X-Tunnels Username
X-Tunnels Password: ****
Use X-Tunnels: Never
Force RTP through X-Tunnels: Default
X-Tunnels challenge Algorithm: MD5
X-Tunnels Encryption: (None)

X-Cipher Host:
X-Cipher Username
X-Cipher Password: ****
Use X-Cipher: Never
Force RTP through X-Cipher: Default
X-Cipher Challenge: SHA1
X-Cipher: AES 128bit

X-Vox Host: xten.net
Use X-Vox: Yes

Advanced System Settings

RTP Settings

Obey Reverse UDP Mapping Rules: No

Send RTCP Messages: Yes

DTMF Force Number: No

DTMF Magic Number: 101

Configuration Settings for X-Lite (Version 3.0)

The settings for version 3.0 have significantly changed from version 2.0. X-Lite 3.0 is typically used with a proxy server and a SIP provider. Here we are simply setting up X-Lite to act as a standalone SIP phone. (If you have a SIP provider, you should follow their instructions to set up your X-Lite phone. In most situations with a SIP provider, you should still be able to dial your HMP box.)

Define a new SIP account by right-clicking on the face of the x-lite and selecting **SIP Account Settings...** from the context menu and then click Add... On the Account Tab, set the following values:

Display Name	(your name)
User Name	(your initials)
Password	(any password will do)
Domain	the ip address of your computer

Ensure the Register with domain is NOT checked. The Dialing plan can remain the same.

Properties of Account1

Account | Voicemail | Topology | Presence | Advanced

User Details

Display Name: Me

User name: Test

Password:

Authorization user name:

Domain: 172.18.13.125

Domain Proxy

Register with domain and receive incoming calls

Send outbound via:

domain

proxy Address:

target domain

Dialing plan: #1\a\A.T;match=1;prestrip=2;

OK Cancel Apply

On the VoiceMail Tab, set the following:

Ensure all check boxes are cleared and that all text boxes are empty except for “0 seconds”.

On the Topology Tab, Set:

IP Address -> Discover Global Address
STUN Server -> Discover Server
Enable ICE (checked)

Manually specify range. If you are going to run X-Lite on the same box as HMP, check this and specify ports 5065-5066, or 5070-5071. If running X-Lite from a different box, you can leave this uncheck and X-Lite will use the default ports 5060-5061.

XTunnels – Never

The screenshot shows the 'Properties of Account1' dialog box with the 'Topology' tab selected. The 'Firewall Traversal' section has 'Discover global address' selected for IP address, 'Discover server' for STUN server, and 'Enable ICE' checked. The 'Port used on local computer' section has 'Manually specify range' checked with the range '5070 - 5071'. The 'XTunnels' section has 'Use XTunnels' set to 'Never', 'Use SIP user name and password' checked, and empty text boxes for 'Server address', 'Username', and 'Password'. The 'OK', 'Cancel', and 'Apply' buttons are at the bottom.

On the Presence Tab:

Mode: Disabled

On the Advanced Tab:

Leave All At the Default

6. Now it is time to make a test call. Save your configuration. Some versions of X-lite require you to press the spacebar to enable alphanumerics. If dialing HMP where both HMP and X-Lite are on the same box, type [Test@xxx.xxx.xxx.xxx:5060](tel:Test@xxx.xxx.xxx.xxx:5060) where xxx represents your IP address. This will call HMP on port 5060.

If calling from a different box (recommended) type [Test@xxx.xxx.xxx.xxx](tel:Test@xxx.xxx.xxx.xxx) where xxx represents the IP address of your HMP box.

7. Click on the Green telephone icon or press enter. It should dial, connect and you should hear a voice prompt. You can terminate the call at any time by clicking on the red telephone icon.

If it is connecting, but you don't hear any audio, this may represent a firewall issue that may not allow SIP and RTP through.

If you wish to test outbound calls from your IVR to the XTen softphone:

When you dial to the softphone, the phone number should be:

If on the same box: [url:ipaddressofmachine:5065](tel:url:ipaddressofmachine:5065)

If on a different box: [url:ipaddressofXlite](tel:url:ipaddressofXlite)

For a test, you can Launch Tmonitor, select on Port 1 and click "Test Dial". Enter [url:ipaddressofmachine:5065](tel:url:ipaddressofmachine:5065) or [url:ipaddressofXlite](tel:url:ipaddressofXlite) and click OK.

You should get a call on your softphone – make sure your microphone works and you say "Hello" as the Test Dial uses call progress and is trying to detect between a HUMAN and MACHINE.

How to use the Cached DLL's

Until now, you had to stop the CTI32 service each time you wanted to install a new CTI32 application. Back in the day of C++ programming directly to the CTI32.DLL, you could simply copy a new application DLL without stopping the system. (as long as there

were no phone calls) The way C++ dynamically linked modules is much different than the way .NET does it. So since converting over to .NET, CTI32 programming has become less convenient – until now.

Here are the steps:

- In the CTI32Config, set the UseDllCache parameter to: true
- Restart the CTI32Engine Service
- Copy in your new DLL anytime you wish to update your code without stopping the engine.

How to enable the Dialogic SIP Logs

When you are using HMP and trying to debug connection problems to a SIP provider, it is often helpful to look at the Intel / Dialogic SIP logs. This section explains how to enable those logs.

In the \program files\intel\hmp\cfg folder, find the file RtfConfigWin.XML file and open it in notepad.

Set the 4th line down to this:

```
<Logfile path="$(INTEL_DIALOGIC_DIR)\log" size="10000" maxbackups="4"/>
```

```
<!-- IP CCLIB GC_H3R SECTIONS -->  
<MLabel name="Debug" state = "1"/>
```

```
<!-- IP CCLIB SIP STACK SECTION -->  
<MLabel name="Debug" state = "1"/>
```

How to improve outbound Dialing Call Progress Results (HMP and DM/V boards)

This section is designed to improve the outbound dialing results of determining when a phone is answered and if it is a human or a machine on the other end. It could also result in a busy, no answer, or invalid number.

This section only applies to DM3 architecture devices such as HMP software or DM/V type boards. For how to improve your outbound dialing results for Springware boards, please review the section below.

You must find the .config file to the protocol that you are running. Go into the Intel DCM, Right click on the board and Configure Device. On the Misc tab, there should be a

description of the FCD File Name and PCD File name. Write the file name down. For example, on my test HMP system it was 4r4v4e4c4s4f4i_host_eva.

In the c:\program files\intel\data folder find the name of the file with the extension .config and edit that file.

Right after
[sigDet]

Add:
init i4

Note: change the i4 to however many channels that you have in your system.

After the 3rd SetQual in the same section add:

!Delete the default PVD qualification template
DeletePvd 128193

!User defined Pvd template.
PvdDesc signalId 128193
PvdDesc signalLabel 0000
PvdDesc minSnr 5
PvdDesc maxSnr 600
PvdDesc maxPk 2
PvdDesc maxRing 5
PvdDesc ringThresh 10000
PvdDesc PvdWin 8
PvdDesc PvdVthresh 250
PvdDesc PvdRbLow 380
PvdDesc PvdRbHigh 510
CreatePvd

!Delete the default PAMD qualification template
DeletePamd 106561

!User defined PAMD template.
PamdDesc signalId 106561
PamdDesc signalLabel 0000
PamdDesc minRing 190
PamdDesc mask 1
PamdDesc maxAnsiz1 125
PamdDesc maxAnsiz2 50
PamdDesc maxAnsiz3 220
PamdDesc loHiss 22
PamdDesc hiHiss 16

```
PamdDesc bhParm 5
PamdDesc cvThresh1 80
PamdDesc cvThresh2 165
PamdDesc maxCvThresh 390
PamdDesc nMaxBroad 2
PamdDesc nMaxErg 65
PamdDesc maxSilence 90
PamdDesc voiceThresh 25
PamdDesc silenceThresh 250
PamdDesc rjFbandLow 0
PamdDesc rjFbandHigh 0
CreatePamd
```

Open a command prompt in the same folder and type the following command:

```
Fcdgen <name of file.config>
```

Then go into the CTI32 Config utility and change the following under the dx_cap section:

```
ca_noanswer - 4000
ca_pamd_failtime - 1000
```

[ca_noanswer](#) increases the time after which a call is considered CR_NOANS to 40 seconds. The default is too short and does not give some voice mail systems enough time to pick up.

[ca_pamd_failtime](#) - increasing this tells the system to allow more time to determine pvd vs pamd. Left at the default, many outbound calls return a status of con_cad because live vs machine can not be determined.

Editing the PCD File

Using a text editor, open the .pcd file corresponding to the .config file you just modified.

1. Scroll down to the [COMP sigdet] section and change the InitOption value from Yes to No. The section should be revised as follows (the change is shown in bold):

```
[COMP sigdet]
```

```
{ Attribute : Std_ComponentType:0x07
  NumInstances : 96
  ConfigOption : YES
  InitOption : NO
  DependentComp : waveAnalyser
```

Note that NumInstances will vary depending on the board in use. In this example, the value reflects a T1 board with 96 channels.

2. Save your changes to the .pcd file and exit.

Stop the CTI32 Service, Stop the Dialogic Service, Start the Dialogic Service, and start the CTI32 Service.

For an explanation of the changes in these parameters. Please refer to the following document:
<http://www.inventivelabs.com/cti32/HMPQualificationTemplates.pdf>

How to improve outbound Dialing Call Progress Results (Springware Boards)

The default Dialogic Perfect Call settings are not that perfect. Expect accuracy in determining a HUMAN or MACHINE about 80%. This can easily be improved into the 90% range.

There are two primary strategies to improving the call progress analysis:

- 1) Use of the Qualification Templates. There is an undocumented feature of Dialogic boards called a qualification template. Many commercial dialers use this feature. This is very easily enabled using the CTI32Config file. To enable this feature, simply set "UseQualificationTemplates" to a threshold parameter of 250.

To understand the settings of the qualification template, please refer to this document and look under the springware section.

<http://www.inventivelabs.com/cti32/HMPQualificationTemplates.pdf>

- 2) Setting the CAP parameters. There are several CAP parameters that can be set to change the behavior of the Perfect Call. For a full explanation of how these parameters work, please refer to this document:

http://resource.intel.com/telecom/support/releases/unix51/linux51fp1/onldoc/htmlfiles/vofeat/1454-03-23.html#P688_32253

Many successful dialers set the CAP parameters in such a way as to get the board to return the so called "Answer Size". The length of the salutation is returned by the GetCapResult(dev,"ATDX_ANSRSIZ") function. (10ms units – 100 = 1 second) By examining the value returned, you can estimate the kind of answer that was received. Normally, a person at home will answer the phone with a brief salutation that lasts about 1 second, such as "Hello" or "Smith Residence." A business will usually answer the phone with a longer greeting that lasts from 1.5 to 3 seconds, such as "Good afternoon, Dialogic Corporation." An answering machine or computer will usually play an extended message that lasts more than 3 or 4 seconds.

Here are some settings that are typical when using this method

```
ca_ansrdgl=50          //Silence of ½ second before terminating Answer Length
ca_maxansr=18000       //Maximum duration of answer.
ca_intflg=6            //Set to this to get Answer Length
```

VALUE OF 6 OR 5 DETECT'S "SIT" & VOICE. THEN RETURNS THE SALUTATION LENGTH IN THE "ATDX_ANSRSIZ" PROPERTY.
VALUE OF 8 DETECTS "SIT" & VOICE THEN RETURNS ; A "0" LENGTH IN THE "ATDX_ANSRSIZ" PROPERTY

```
ca_hedge=2            // Edge of answer to send connect message.
ca_cnosig=4000        //Duration of no signal time out delay.
ca_noanswer=4000      //time before no answer after 1st ring
ca_maxintering=1200  // Max inter ring delay before connect
```

All of these settings can be modified in the CTI32 Config utility under the DX_CAP section.

Using the DnisMethod

A powerful feature of CTI32 is the ability to run multiple applications concurrently on a single computer depending on the dialed phone number (DNIS). These applications can all run within on “class library” module or they could each run from a separate DLL.

Here's how it works:

You set up the method name in the CTI32 config utility. Set the DnisMethod to the name of the special DNIS handler in the DefaultDLL.

Here is a sample C# DnisMethod function:

```
//The idea behind this method is to return to the Engine which Module
and Method should be loaded
//given the dialed number.
public int DnisLogic(ConfigData cfg,ChannelData lineData,String
dnis,StringBuilder module,StringBuilder method, StringBuilder
classType)
{
    log=cfg.log;
    log.Write("DnisLogic: Finding module for DNIS {0}",dnis);

    //Look up the DNIS from the database
    module.Append("Dialer.DLL");
    method.Append("InboundCall");
    classType.Append("Dialer.Dialer");
    return (0);
}
```

You could use a database to based on the DNIS and configure which module to call.

Once you know which application to run, based on the dialed phone number, you simply load up the DLL name, the method to call and the class type.

If you return 0, the call will proceed. If you return a non-zero return, then the call is dropped.

Getting the TTS Functions to work

CTI32 relies on the Microsoft SAPI engine being installed. This may already be installed on XP or 2003 systems (or newer). You can check to see if the engine is installed by going to control panel and clicking on Speech. If you can test the TTS function, then you have SAPI installed. Otherwise download the Speech 5.1 SDK from Microsoft.

<http://www.microsoft.com/downloads/details.aspx?FamilyID=5e86ec97-40a7-453f-b0ee-6583171b4530&DisplayLang=en>

Download and install SpeechSDK51.EXE

You should now have a speech icon in control panel. Click on the TTS tab and test an installed voice.

You must set up the CTI32.INI file to enable TTS. There should be an entry that looks similar to this:

```
[TTS]
Name=Microsoft Mary
InitializeOnStart=1
VoiceFormat=1

;Valid VoiceFormats: (Defaults to 6)
;1 - Kbps32 - ADCPM 8Khz
;2 - Kbps64 - PCM 8Khz 8-bit mono
;5 - Wave8 - PCM 8Khz 8-bit Mono
;6 - Wave11 - PCM 11Khz 8-bit Mono
;8 - GSM610
;10 - Wave8Mulaw - Mulaw 8Khz 8bit Mono
;11 - Wave11Mulaw - Mulaw 11Khz 8bit Mono
;12 - Wave8Alaw - Alaw 8Khz 8bit Mono
;13 - Wave11Alaw - Alaw 11Khz 8bit Mono
;14 - Vox8Mulaw - Mulaw 8Khz 8bit Mono
;Note: use Vox8Mulaw (14) if you are going to use the PlayTTSAndRecognizeWord
function
```

If you use ATT Natural Voices, the entry might look like this:

[TTS]

Name=ATT DTNV 1.4 Mike16

InitializeOnStart=1

The name of the voice should appear in the drop down on the control panel / speech / Text to Speech tab.

You can always check the CTI32.Log file to see the results of the Sapi initialize functions. Search for (T).

HMP Third Party Call Control Walkthrough

As explained in CTI32.dll.doc, third party call control gives you greater flexibility in processing RTP streams in a sip environment. Assume that you have received a SIP call and have routed the call to another SIP call over the software CT bus. In this case you are handling the RTP streams for both calls. This is taking up bandwidth and utilizing two IP Media devices on your system.

In this example, what we would like to do is get the two sip devices to send the RTP streams to each other instead of hair-pinning them through the server. You still want to maintain call control to know when the parties hang up, and to regulate the amount of call time the two parties talk. You may also want to get the streams back at a later time.

The major steps involved are:

Get SDP information on the established calls:

```
rc = cti.GetSipStats(party1.tldev, ref codec1, ref address1, ref port1, ref sdp1);
```

```
rc = cti.GetSipStats(party2.tldev, ref codec2, ref address2, ref port2, ref sdp2);
```

Each caller is re-invited separately. Here we ask party1 to send its RTP stream to party2's address and port, using the codec it is already using, and, if successful, asking CTI32 to release the media stream resource.

```
rc = cti.SipReinvite(party1.tldev, codec2, address2, port2, 1, "");
```

If the first party rejects the re-invoke, then you can keep the call going through the local server. If the first party accepts, you must also ask the second party to switch:

```
rc = cti.SipReinvite(party2.tldev, codec1, address1, port1, 1, "");
```

If the second party rejects the re-invoke after the first has accepted, you need to get the first party back:

```
if (rc !=0)
{
    rc = cti.SipReinvite(party1.tldev, codec1, "localhost", 0, 0, sdp1);
}
```

If you can't get the first party back, then you must terminate the call on both ends. As party1 is transmitting RTP data into the void.