Dialogic®

Application Note

Profiling a High Density/Frequent Switching Conferencing Application

Executive Summary

This application note provides information regarding a study of a high density, high performance conferencing application utilizing Dialogic® HMP Software and Dialogic® HMP DNI Boards. The conferencing application was written using the CNF API, and the hardware system comprised up to five Dialogic® DNI/1200TEPHMP Digital Network Interface Boards. The host system was the Alliance I-2000 R5 Server in combination with the Alliance EX-4000 Expansion Platform. This application note profiles the system's performance using a sample application that showcases 500 active calls inbound to a conference server from a mix of IP and PSTN based endpoints.

Table of Contents

Introduction

Dialogic® HMP Software includes support for the CNF API. The CNF API is an asynchronous conferencing API that can be used to construct large scale conferencing applications. The CNF API's asynchronous design allows for faster switching times and higher density than was possible with previous conferencing APIs (for example, MSI and DCB).

Note: The DCB API, while available on the Dialogic HMP Software, is deprecated and new features will be added on the CNF conferencing API.

The application note profiles the capabilities of the CNF API for handling:

1. Large Conference Participant density
2. Frequent conference switches
3. Input from both IP and PSTN Thin blades
4. Voice quality of the transcoding, summation, and echo cancellation algorithms

To measure the CNF API capabilities, the following was tracked:

1. The time it takes for one party to move from Conf A to Conf B
2. The CPU Utilization of the system under test for the density
3. The impact of the context switching and density on other threads present in the same process
4. Voice Quality spot checks. This performance point validates the quality of the audio to make sure that there is no distortion from EC or Summation on the conference. In some cases, some audio anomalies were caused by network corruption. When this was the case, it is noted. Test runs on a LAN better tuned to VoIP could eliminate these network issues.

Sample Application Overview

Two main applications were used in the profiling (samples of this code can be downloaded [see the *For More Information* section]):

1. **GCBCM** — This application is based on the gc_basic_call_model demo that is provided as part of the system release. As a modification, post connect voice functionality was added to the outbound_application function. This voice functionality was used to simulate inbound callers into the conference servers by plays, records, and digit functions.
2. **CNFTest** — This conference server application was developed for this testing. The CNFTest uses the CNF API to create conferences and parties. It uses Dialogic® Global Call Software to accept the IP and PSTN calls. The application will take several command line arguments to adjust the conference party count, conference count, and timing profiles. Also, this application provides extensive logging that was used to calculate the performance data.

Running the CNFTest Application

The CNFTest has the following command line arguments:

-?	prints the help
-bd	sets the CNF board name, default=cnfB1
-p	sets the maxnumber of parties, default=2
-c	sets the maxnumber of CONF devices, default=1
-l	sets the log level, Levels are: [ERROR,EVENT,API,INFO,ENTRY,EXIT,DEBUG,ALL], default=ALL

-mode	sets the SRL event processing mode (SR_MTASYNC(default),SR_STASYNC,SR_POLLMODE)
-a	sets the application to switch announce mode where in between party conf switches the file in –f is played
-f	sets the voice playfile, default=playfile.vox
-h	sets SRL Heartbeat on, default=0 (off)
-t	sets the switchtime, default=5000 (5sec)
-rt	sets the switchtime to random, default=fixed
-rc	sets the conf to random, default=roundrobin
-dti	sets to run in DTI only mode routing no frontend, default=vox mode
-vox	sets to run in VOX only mode routing no frontend, default=vox mode
-gc	sets the number of GC DTIFrontend to open default=0
-ip	sets the number of GC IPfrontend to open default=0
-dc	sets it to do devconnect each time a party changes conferences
-cb	sets the time the Callback handler should pause before returning, set to -1 to disable. Default=-1
-q	sets the exit timeout in seconds, default=never

A typical test run would usually require setting –p, -c, -t, -dti/vox, or –ip/gc.

To accept 500 inbound calls (250IP and 250PSTN) into 50 different conferences, would look as follows:

```
CNFTest –p 500 –c 50 –gc 250 –ip 250
```

Application Call Flow

1. **Wait for Inbound Call event** — For this case, we wait for the inbound call to be connected. For the Global Call calls (either IP or PSTN), the call is connected, then the application posts the user-defined USREV_CALLCONNECTED event to indicate that the call is connected. Stimuli calls attempt to reconnect to the system immediately upon disconnect.

2. **Route Inbound Call to Party Device** — When the inbound call is received, the application routes the front-end device via the dev_connect API. This causes two DMEV_CONNECT events to be generated, one for each leg of the full duplex route. In this case, the event for the front-end device is ignored, and only state change is executed off the Party event.

3. **Add party into conference** — When the party event is received, the party device is added into a conference. This may be a fixed conference assignment or placed into a random conference (based on command line argument). When this add to conference completes, it causes the CNFEV_ADD_PARTY event to be generated. At this point the conferee is inside the conference and the audio flow is bidirectional. The application then starts a switch timer (based on command line argument).

4. **Wait for switch timer event** — When the switch timer completes, the application posts the user-defined USREV_SWITCHTIMER event. At this time the application removes the party device from the conference. If the –a option is set, the application plays a voice prompt (causing the TDX_PLAY to be generated). If the –dc is set, it unroutes the party dev and frontend, and reroutes it; otherwise, the application just goes back to step 3 and adds the party to another conference.

   From this point, the application will loop through moving the party devices to different conferences. The stimulus application is responsible for disconnecting all calls.

5. **Disconnect the call** — The call is disconnected after the far end hangs up.

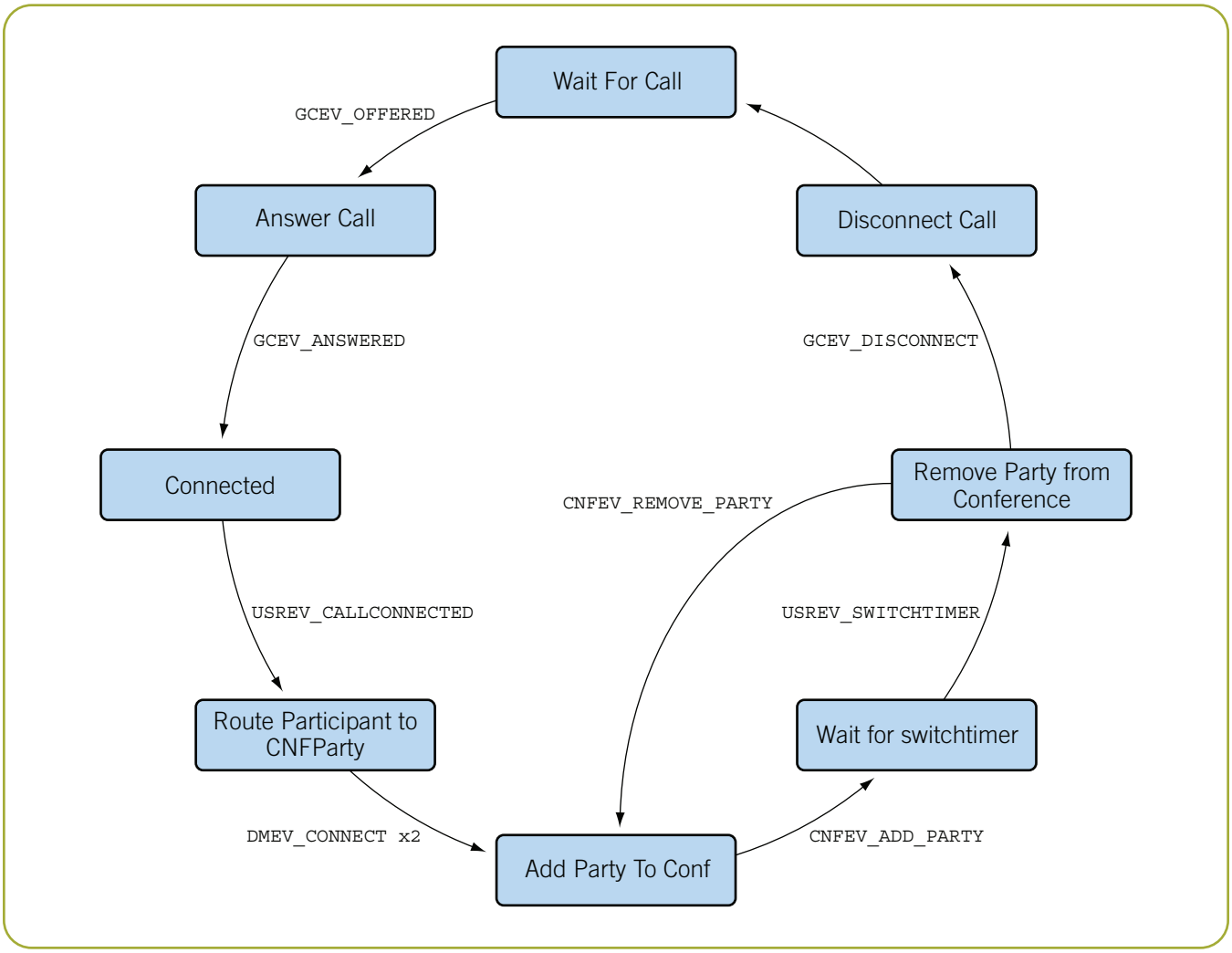

Figure 1. Application Call Flow

Application Performance Data

1. **Function Timers** — This performance point is the time each function takes from calling to the point where the application receives the termination event.
2. **Party Switch Time** — This performance point tracks the amount of time it takes for party to go from hearing audio from one conference to another.

   Note: If the –a is enabled, this switch time includes the full length of the play. If –dc is enabled, it includes the dev connects and disconnects. Also, the Switch Time standard deviation is listed so that the range of switch time is understood.
3. **SRL Throughput** — The application has the ability to turn on the SRL Heartbeat. When this is turned on, a thread places a user-defined USREV_SRLHEARTBEAT event onto the SRL event queue and tracks the amount of time it takes to be processed by the process Event function. This functionality allows the application to trace the health of the SRL event queue. The SRL Throughput is used to see if the SRL models are capable of moving through the event density that is being created in the call and switch events.

   Note: This is the baseline latency for all events. Subtracting this time from the function times gives a full picture of the time a given function is taking.

4. **Process Event Timer** — This is the amount of time spent inside the process event function.
5. **String Replace Time and Count** — This performance point is monitoring the performance characteristics of a separate application thread that is performing some fixed task. This performance data is to understand the impact (if any) of non-Dialogic processing tasks inside the system.
6. **Host Utilization Data** — Performance data is measured by perfmon and process monitor including CPU utilization and network saturation count.
7. **Voice Quality Review** — Some of the stimulus channels randomly do records rather than plays. The record files are then harvested and inspected for voice quality anomaly. Similarly, some of the inbound calls are made from IPSoftphones and the quality listened to.

Test Results from Test Runs

Test 1: Baseline Test, VOX

Description

This is a basic functionality test where voice devices are opened and used as the party devices inside a conference. This test took baseline performance data, as there is no call control or external box routing. In this test, the VOX devices were idle so there was no audio data, just testing switching throughput.

System Setup

Configuration 1 — System under test:

Windows® 2003 Svr R2
Dual 3.8 Xeon with HyperThreading enabled (8 virtual processors)
2 GB Memory
Gigabit Ethernet
Dialogic® Host Media Processing Software Release 3.0 for Windows® B144
CNFTest Application

For this configuration no outside stimulus was needed.

Results

1 channel (-p 1 –c 1 –t 4000 –rt –rc -vox)

Profile Point	Result (in ms)
Function Timer	18
Party Switch Timer	31
Party Switch Time Standard Deviation	0.1
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70058)
Host Utilization Data	11%
Voice Quality	NA

10 channel (-p 10 –c 5 –t 4000 –rt –rc -vox)

Profile Point	Result (in ms)
Function Timer	19
Party Switch Timer	33
Party Switch Time Standard Deviation	0.2
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70024)
Host Utilization Data	13%
Voice Quality	NA

100 channel (-p 100 –c 50 –t 4000 –rt –rc -vox)

Profile Point	Result (in ms)
Function Timer	19
Party Switch Timer	34
Party Switch Time Standard Deviation	0.7
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (69366)
Host Utilization Data	16%
Voice Quality	NA

250 channel (-p 250 –c 150 –t 4000 –rt –rc -vox)

Profile Point	Result (in ms)
Function Timer	19
Party Switch Timer	35
Party Switch Time Standard Deviation	.9
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (68534)
Host Utilization Data	19%
Voice Quality	NA

500 channel (-p 500 –c 250 –t 4000 –rt –rc -vox)

Profile Point	Result (in ms)
Function Timer	22
Party Switch Timer	42
Party Switch Time Standard Deviation	1.3
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (67097)
Host Utilization Data	21%
Voice Quality	NA

500 channel MultiConnect (-p 500 –c 250 –t 4000 –rt –rc –dc -vox)

Profile Point	Result (in ms)
Function Timer	27
Party Switch Timer	79
Party Switch Time Standard Deviation	1.3
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (66871)
Host Utilization Data	25%
Voice Quality	NA

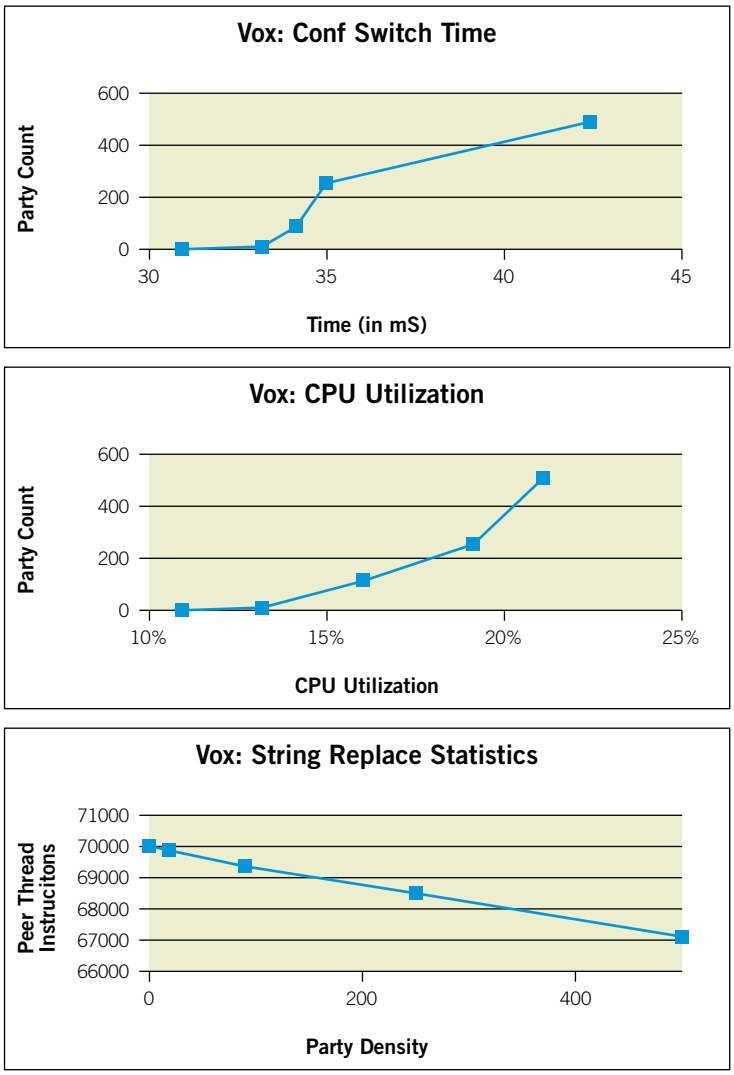

Figure 2. Test 1: Baseline Test, VOX

Test 2: Baseline Test, DTI

Description

This is a basic functionality test where DTI devices are opened and used as the party devices inside a conference. This test took baseline performance data, as there is no call control or external box routing. In this test, the DTI devices were idle so there was be no audio data, just testing switching throughput.

System Setup

Configuration 1 — System under test:

Windows® 2003 Svr R2
Alliance Systems I-2000 R5 Server
Dual 3.6 5100 Xeon (4 virtual processors)
2 GB Memory
Gigabit Ethernet
Dialogic HMP Software 3.0 B144
Dialogic® DNI/1200TEPHMP Digital Network Interface Board connected back-to-back running NI2 ISDN
Alliance Systems EX-4000 Expansion Platform
CNFTest Application

For this configuration no outside stimulus was needed.

Results

1 channel (-p 1 –c 1 –t 4000 –rt –rc -dti)

Profile Point	Result (in ms)
Function Timer	16
Party Switch Timer	24
Party Switch Time Standard Deviation	.1
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70274)
Host Utilization Data	9%
Voice Quality	NA

23 channel (1 span on DNI/1200TEPHMP) (-p 23 –c 10 –t 4000 –rt –rc -dti)

Profile Point	Result (in ms)
Function Timer	17
Party Switch Timer	25
Party Switch Time Standard Deviation	1.3
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70121)
Host Utilization Data	11%
Voice Quality	NA

92 channel (1 DNI/1200TEPHMP) (-p 92 –c 40 –t 4000 –rt –rc -dti)

Profile Point	Result (in ms)
Function Timer	17
Party Switch Timer	29
Party Switch Time Standard Deviation	1.2
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70008)
Host Utilization Data	13%
Voice Quality	NA

184 channel (2 DNI/1200TEPHMP) (-p 184 –c 90 –t 4000 –rt –rc -dti)

Profile Point	Result (in ms)
Function Timer	19
Party Switch Timer	34
Party Switch Time Standard Deviation	2.8
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (69992)
Host Utilization Data	17%
Voice Quality	NA

460 channel (5 DNI/1200TEPHMP) (-p 460 –c 250 –t 4000 –rt –rc -dti)

Profile Point	Result (in ms)
Function Timer	21
Party Switch Timer	38
Party Switch Time Standard Deviation	3.1
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (68461)
Host Utilization Data	23%
Voice Quality	NA

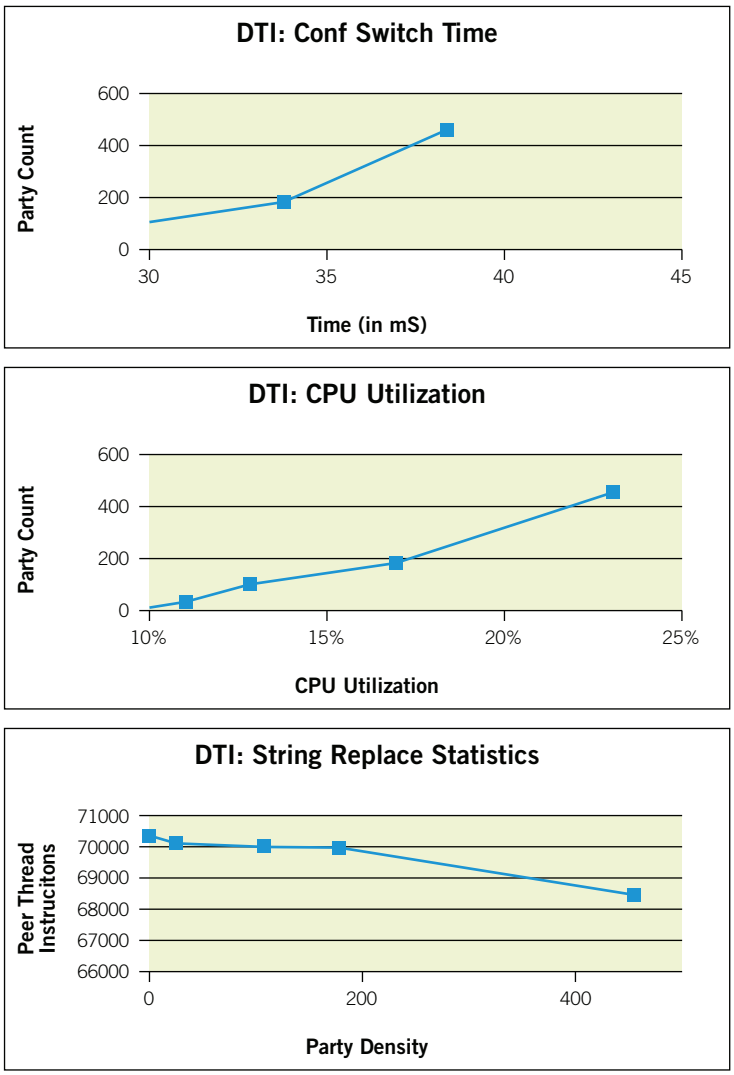

Figure 3. Test 2: Baseline Test, DTI

Test 3: IP Test

Description

This is a basic functionality test where IP calls are made to the box. Each test call was placed with G.711 20 ms encoding. On the stimulus side, one-fifth of the calls made recordings and four-fifths played unique files. During the one-hour test runs, recordings were sampled and examined for audio quality.

System Setup

Configuration 1 — System under test:

- Windows® 2003 Svr R2
- Alliance Systems I-2000 R5 Server
- Dual 3.6 5100 Xeon (4 virtual processors)
- 2 GB Memory
- Gigabit Ethernet
- Dialogic HMP Software 3.0 B144
- DNI/1200TEPHMP connected back-to-back running NI2 ISDN
- Alliance Systems EX-4000 Expansion Platform
- CNFTest Application

Configuration 2 — Stimulus:

- Windows® 2003 Svr R2
- Dual 3.8 Xeon with HyperThreading enabled (8 virtual processors)
- 2 GB Memory
- Gigabit Ethernet
- Dialogic HMP Software 3.0 B144
- GCBCM 4/5 Play 1/5 Record

Results

1 channel (-p 1 –c 1 –t 4000 –rt –rc –ip 1)

Profile Point	Result (in ms)
Function Timer	18
Party Switch Timer	31
Party Switch Time Standard Deviation	2.6
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70158)
Host Utilization Data	11%
Voice Quality	NA

10 channel (-p 10 –c 5 –t 4000 –rt –rc –ip 10)

Profile Point	Result (in ms)
Function Timer	19
Party Switch Timer	32
Party Switch Time Standard Deviation	2.3
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70045)
Host Utilization Data	13%
Voice Quality	20/20 checks pass

100 channel (-p 100 –c 50 –t 4000 –rt –rc –ip 100)

Profile Point	Result (in ms)
Function Timer	20
Party Switch Timer	42
Party Switch Time Standard Deviation	3.1
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (68343)
Host Utilization Data	16%
Voice Quality	50/50 checks pass

250 channel (-p 250 –c 150 –t 4000 –rt –rc –ip 250)

Profile Point	Result (in ms)
Function Timer	22
Party Switch Timer	47
Party Switch Time Standard Deviation	3.8
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (67534)
Host Utilization Data	19%
Voice Quality	49/50 checks pass

Note: Voice quality failure had 3-4 small clicks present in the recordings

500 channel (-p 500 –c 250 –t 4000 –rt –rc –ip 500)

Profile Point	Result (in ms)
Function Timer	23
Party Switch Timer	51
Party Switch Time Standard Deviation	3.4
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (66088)
Host Utilization Data	27%
Voice Quality	48/50 checks pass

Note: Voice quality issues seem to be related to IP traffic issues rather than summation or Echo cancellation issues. The voice anomalies were very small clicks or dropouts 1-2 packets in length.

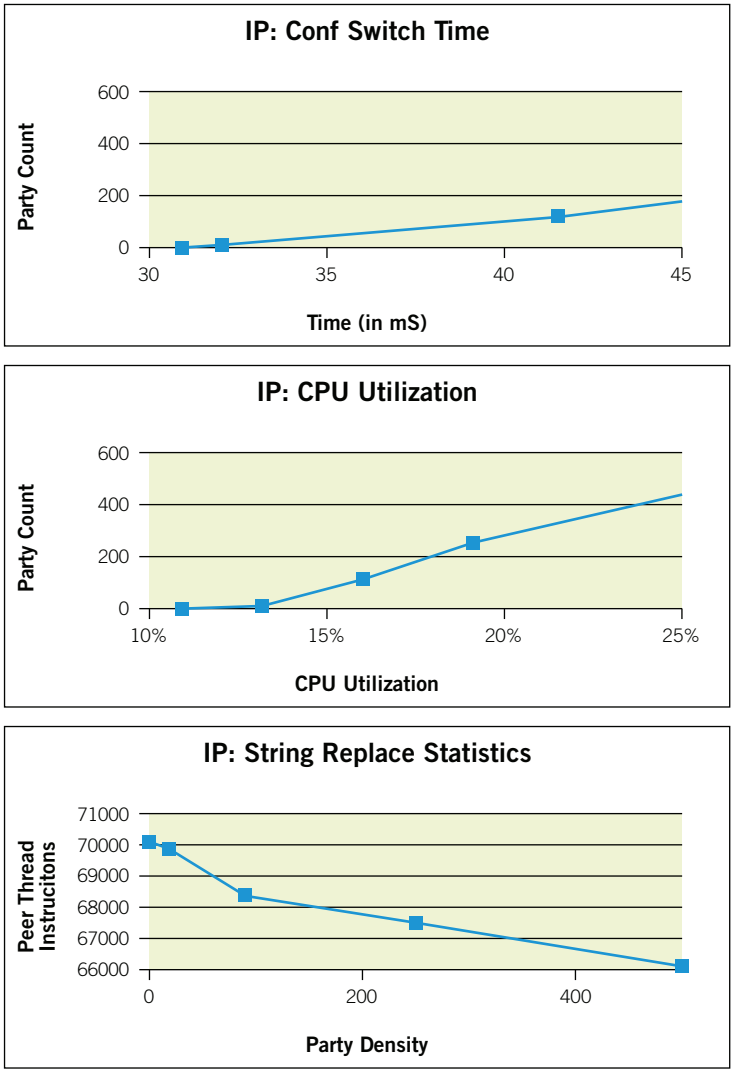

Figure 4. Test 3: IP Test

Test 4: PSTN Test

Description

This is a basic functionality test where PSTN calls are made to the box. Each test call was placed with NI2 ISDN. On the stimulus side, one-fifth of the calls made recordings and four-fifths played known files. During the one-hour test runs, recordings were sampled at random and examined for audio quality.

System Setup

Configuration 1 — System under test:

Windows® 2003 Svr R2
Alliance Systems I-2000 R5 Server
Dual 3.6 5100 Xeon (4 virtual processors)
2 GB Memory
Gigabit Ethernet
Dialogic HMP Software 3.0 B144
DNI/1200TEPHMP connected back-to-back running NI2 ISDN
Alliance Systems EX-4000 Expansion Platform
CNFTest Application

Configuration 2 — Stimulus:

Windows® XP Pro
P4 2.6G
1 GB Memory
Dialogic® System Release Software 6.0 PCI for Windows® Service Update 160
5 x DM/V960A
GCBCM 4/5 Play 1/5 Record

Results

1 channel (-p 1 –c 1 –t 4000 –rt –rc –gc 1)

Profile Point	Result (in ms)
Function Timer	16
Party Switch Timer	24
Party Switch Time Standard Deviation	.8
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70233)
Host Utilization Data	9%
Voice Quality	NA

23 channel (1 span on DNI/1200TEPHMP) (-p 23 –c 10 –t 4000 –rt –rc –gc 23)

Profile Point	Result (in ms)
Function Timer	17
Party Switch Timer	25
Party Switch Time Standard Deviation	.8
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70095)
Host Utilization Data	11%
Voice Quality	20/20 Pass

92 channel (1 DNI/1200TEPHMP) (-p 92 –c 40 –t 4000 –rt –rc –gc 92)

Profile Point	Result (in ms)
Function Timer	17
Party Switch Timer	29
Party Switch Time Standard Deviation	1.7
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (69987)
Host Utilization Data	13%
Voice Quality	50/50 Pass

184 channel (2 DNI/1200TEPHMP) (-p 184 –c 90 –t 4000 –rt –rc –gc 184)

Profile Point	Result (in ms)
Function Timer	19
Party Switch Timer	35
Party Switch Time Standard Deviation	1.4
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (69791)
Host Utilization Data	19%
Voice Quality	50/50 Pass

460 channel (5 DNI/1200TEPHMP) (-p 460 –c 250 –t 4000 –rt –rc –gc 460)

Profile Point	Result (in ms)
Function Timer	22
Party Switch Timer	41
Party Switch Time Standard Deviation	1.5
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (69666)
Host Utilization Data	24%
Voice Quality	50/50 Pass

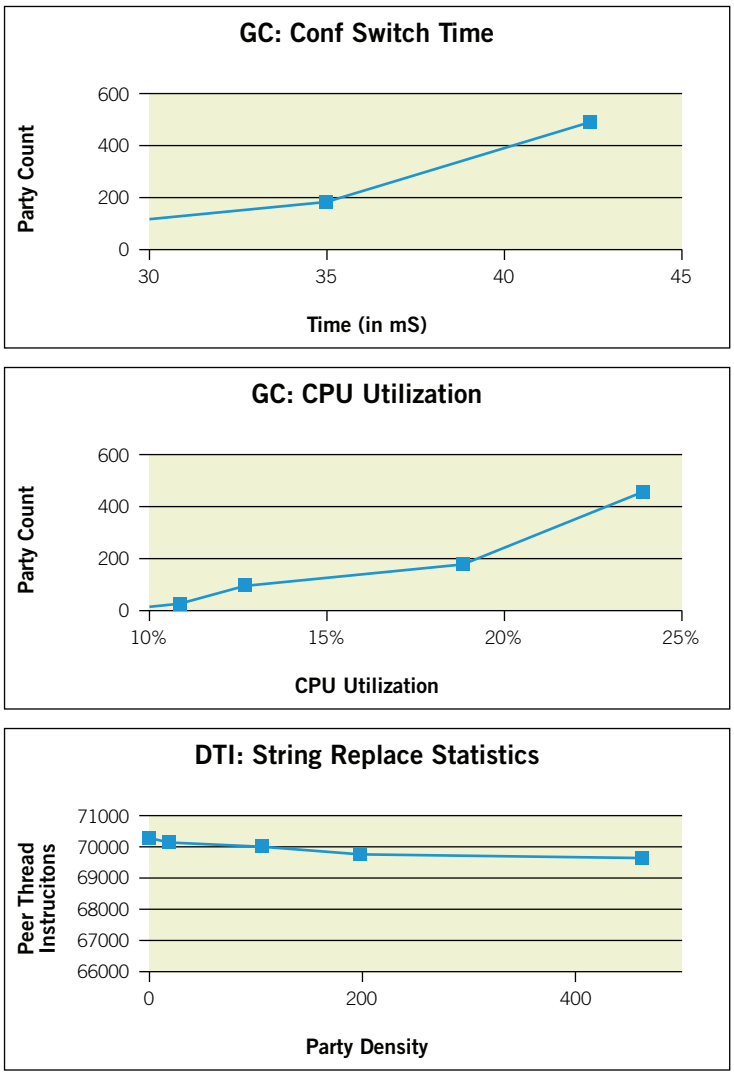

Figure 5. Test 4: PSTN Test

Test 5: Hybrid Test

Description

This is a basic functionality test where PSTN and IP calls are made to the system under test. Each PSTN test call was placed with NI2 ISDN and each IP call was made with G.711 20 ms. On the stimulus side, one-fifth of the calls made recordings and four-fifths played known files. During the one-hour test runs, recordings were sampled at random and examined for audio quality.

System Setup

Configuration 1 — System under test:

Windows® 2003 Svr R2
Alliance Systems I-2000 R5 Server
Dual 3.6 5100 Xeon (4 virtual processors)
2 GB Memory
Gigabit Ethernet
Dialogic HMP Software 3.0 B144
DNI/1200TEPHMP connected back to back running NI2 ISDN
Alliance Systems EX-4000 Expansion Platform
CNFTest Application

Configuration 2 — Stimulus:

Windows® XP Pro
P4 2.6G
1 GB Memory
System Release 6.0 PCI Windows Service Update 160
5 x DM/V960A
GCBCM 4/5 Play 1/5 Record

Configuration 3 — Stimulus:

Windows® 2003 Svr R2
Dual 3.8 Xeon with HyperThreading enabled (8 virtual processors)
2 GB Memory
Gigabit Ethernet
Dialogic HMP Software 3.0 B144
GCBCM 4/5 Play 1/5 Record

Results

2 channel (-p 2 –c 1 –t 4000 –rt –rc –gc 1 –ip 1)

Profile Point	Result (in ms)
Function Timer	16
Party Switch Timer	25
Party Switch Time Standard Deviation	2.1
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70251)
Host Utilization Data	9%
Voice Quality	NA

50 channel (1 span on DNI/1200TEPHMP) (-p 50 –c 20 –t 4000 –rt –rc –gc 23 –ip 27)

Profile Point	Result (in ms)
Function Timer	21
Party Switch Timer	32
Party Switch Time Standard Deviation	1.7
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (70195)
Host Utilization Data	13%
Voice Quality	20/20 Pass

200 channel (1 DNI/1200TEPHMP) (-p 200 –c 75 –t 4000 –rt –rc –gc 92 –ip 108)

Profile Point	Result (in ms)
Function Timer	17
Party Switch Timer	39
Party Switch Time Standard Deviation	2.0
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	31 ms (69892)
Host Utilization Data	19%
Voice Quality	50/50 Pass

500 channel (3 DNI/1200TEPHMP) (-p 500 –c 250 –t 4000 –rt –rc –gc 276 –ip 234)

Profile Point	Result (in ms)
Function Timer	26
Party Switch Timer	44
Party Switch Time Standard Deviation	3.5
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (67673)
Host Utilization Data	25%
Voice Quality	50/50 Pass

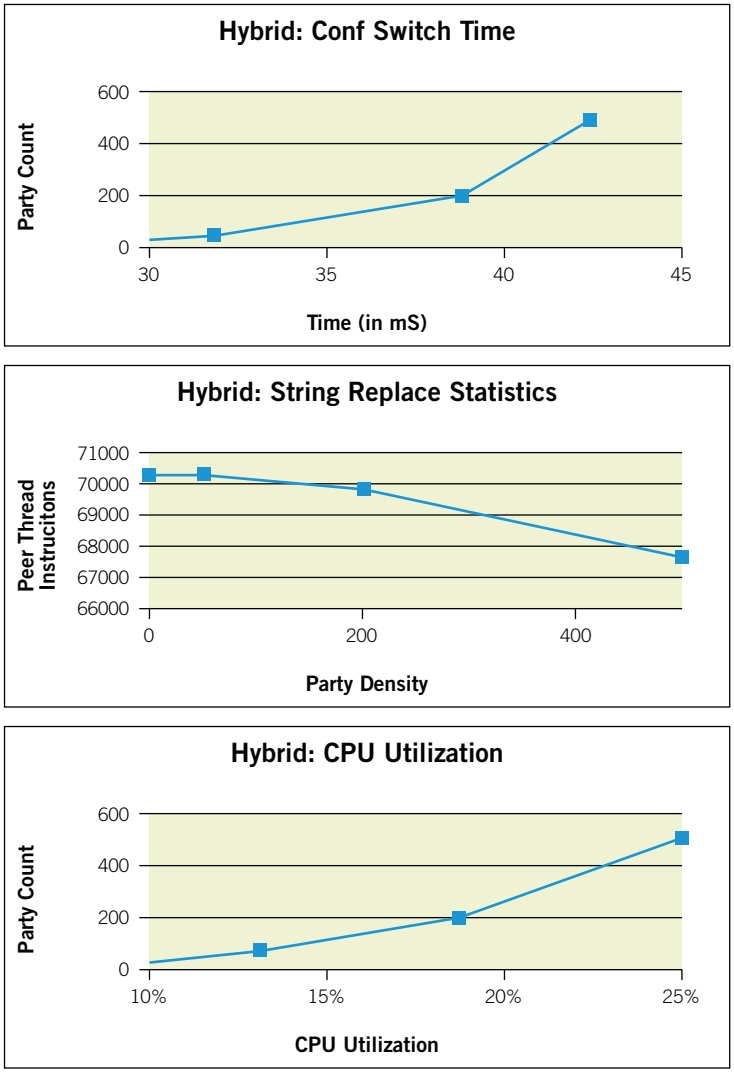

Figure 6. Test 5: Hybrid Test

Summary of Results from Tests 1 through 5

Switch Time

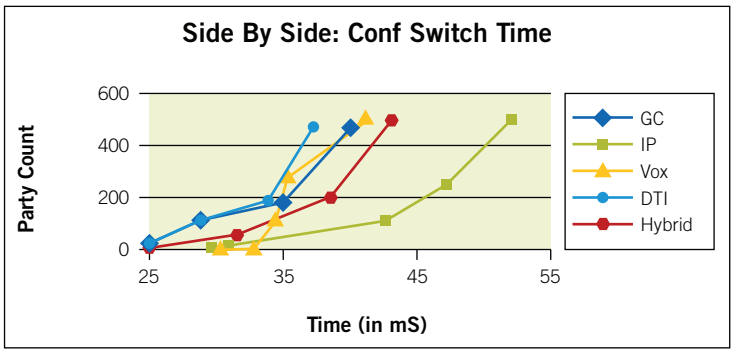

Figure 7. Summary of Results from Tests 1 through 5

For all test runs, the switch time seemed to roughly scale with the port density. The historic APIs add some latency to the switches that go through these APIs. These are eliminated on the IP implementation via the use of the dev_connect API. Though even in maximum density, the switch time is well within the 75 to 100 ms threshold that is typical for this kind of application.

CPU Utilization

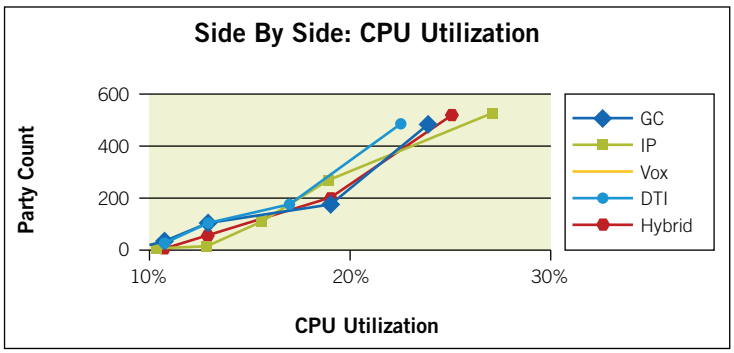

Figure 8. CPU Utilization

The CPU utilization for the CNF portion of this application remains both fixed and roughly linear as the party density increases.

String Replace Count

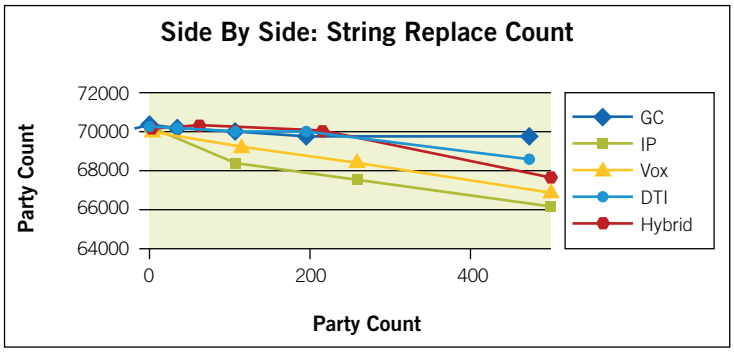

Figure 9. String Replace Count

This chart indicates that the impact of the multiple channels has roughly the same impact despite the technology used. IP does seem to incur some extra overhead per channel, but it scales in the same fashion as the other technologies.

Test 6: Announce Prompt Test

Description

These test runs used the announce prompt in between the conference switch. This plays a two-second prompt that says "Now changing conferences". To do this, there are several different routes to VOX devs and back. Also, there is some additional overhead on the box because of the voice plays.

As before, there were PSTN and IP calls are made to the box. Each PSTN test call was placed with NI2 ISDN and each IP call was made with G.711 20 ms. On the stimulus side, one-fifth of the calls made recordings and four-fifths played known files. During the one-hour test runs, recordings were sampled at random and examined for audio quality.

Note: The conference switch times included the 2000 ms for the two-second announce prompt to be played.

System Setup

Configuration 1 — System under test:

Windows® 2003 Svr R2
Alliance Systems I-2000 R5 Server
Dual 3.6 5100 Xeon (4 virtual processors)
2 GB Memory
Gigabit Ethernet
Dialogic HMP Software 3.0 B144
DNI/1200TEPHMP connected back to back running NI2 ISDN
Alliance Systems EX-4000 Expansion Platform
CNFTest Application

Configuration 2 — Stimulus:

Windows® XP Pro
P4 2.6G
1GB Memory
System Release 6.0 PCI Windows Service Update 160
5 x DM/V960A
GCBCM 4/5 Play 1/5 Record

Configuration 3 — Stimulus:

Windows® 2003 Svr R2
Dual 3.8 Xeon with HyperThreading enabled (8 virtual processors)
2 GB Memory
Gigabit Ethernet
Dialogic HMP Software 3.0 B144
GCBCM 4/5 Play 1/5 Record

500 channel (AllIP) (-p 500 –c 250 –t 4000 –rt –rc –ip 500 -a)

Profile Point	Result (in ms)
Function Timer	31
Party Switch Timer	101
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (67401)
Host Utilization Data	33%
Voice Quality	47/50 Pass

Voice quality failures were similar to the other test run. They seemed to be network-induced due to additional traffic for IP calls. Test runs on a network that was tuned and allocated for this level of IP traffic yielded better results (100/100 in testing), but that network was not available for general testing of this configuration.

Note: The conference switch times included the 2000 ms for the two-second announce prompt to be played, but that time was removed for purposes of comparison on this chart.

500 channel (3 DNI/1200TEPHMP) (-p 500 –c 250 –t 4000 –rt –rc –gc 276 –ip 234 -a)

Profile Point	Result (in ms)
Function Timer	31
Party Switch Timer	101
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (67401)
Host Utilization Data	31%
Voice Quality	49/50 Pass

Voice quality failures were similar to the other test runs. They seemed to be network-induced due to additional traffic for IP calls.

Note: The conference switch times included the 2000 ms for the 2 second announce prompt to be played, but that time was removed purposes of comparison on this chart.

Test 7: Large Conference Density

Description

The test runs tested large conference participant/conference ratios in the 10 to 20 parties per conference range. Previous test runs had 2 to 4 parties per conference. Due to the number of participants and the "noise" of the play files, it was too difficult to distinguish the prompts to test for voice quality.

As before, PSTN and IP calls were made to the box. Each PSTN test call was placed with NI2 ISDN, and each IP call was made with G.711 20 ms. On the stimulus side, one-fifth of the calls made recordings and four-fifths played known files. During the one-hour test runs, recordings were sampled at random and examined for audio quality.

System Setup

Configuration 1 — System under test:

Windows® 2003 Svr R2
Alliance Systems I-2000 R5 Server
Dual 3.6 5100 Xeon (4 virtual processors)
2 GB Memory
Gigabit Ethernet
Dialogic HMP Software 3.0 B144
DNI/1200TEPHMP connected back to back running NI2 ISDN
Alliance Systems EX-4000 Expansion Platform
CNFTest Application

Configuration 2 — Stimulus:

Windows® XP Pro
P4 2.6G
1GB Memory
System Release 6.0 PCI Windows Service Update 160
5 x DM/V960A
GCBCM 4/5 Play 1/5 Record

Configuration 3 — Stimulus

Windows® 2003 Svr R2
Dual 3.8 Xeon with HyperThreading enabled (8 virtual processors)
2 GB Memory
Gigabit Ethernet
Dialogic HMP Software 3.0 B144
GCBCM 4/5 Play 1/5 Record

500 channel (AllIP) (-p 500 –c 40 –t 4000 –rt –rc –ip 500)

Profile Point	Result (in ms)
Function Timer	21
Party Switch Timer	44
Party Switch Time Standard Deviation	6.3
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (66401)
Host Utilization Data	26%
Voice Quality	NA

500 channel (3 DNI/1200TEPHMP) (-p 500 –c 40 –t 4000 –rt –rc –gc 276 –ip 234)

Profile Point	Result (in ms)
Function Timer	17
Party Switch Timer	38
Party Switch Time Standard Deviation	2.8
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (67228)
Host Utilization Data	25%
Voice Quality	NA

500 channel long conf timer (3 DNI/1200TEPHMP) (-p 500 –c 40 –t 300000 –rt –rc –gc 276 –ip 234)

Profile Point	Result (in ms)
Function Timer	16
Party Switch Timer	36
Party Switch Time Standard Deviation	4.4
SRL Heartbeat Timer	0
Process Event CB Timer	0
SR Time and count	32 ms (68628)
Host Utilization Data	27%
Voice Quality	NA

High Performance Application Designs

This section discusses some observations to consider when attempting to implement this sort of high scale conferencing application.

Event Density and Processing Throughput

Because of the highly asynchronous nature of the CNF API, one potential cause for delay points inside the application comes from the number of events generated by a single caller and the amount of time needed to process each event. If events are being generated at a faster rate than the application is processing them, the delay times for switching will start to increase and the system will start to feel sluggish.

To calculate this, consider the following regarding the application and the use case of the parties:

1. Average number of callers concurrently using the system
2. Number of events generated by each caller, including:

    Events for call control to establish and tear down the call

    Events used to direct participants to and from conference (IVR events, Digits, etc.)
3. Average time spent in each conference (that is, frequency of movements)
4. Average time the participant is connected to server
5. Average time to process a single event

We can profile the amount of time it takes to process an event by tracking time between sr_waitevt calls or the time spent inside the callback function.

With these pieces of information, we can calculate the number of events that are being generated in the system, and provide that the number of events is significantly less than the amount of time it takes to process a single event.

Example of Calculation

In this application, we generate the following for each caller:

Call Control events

GCEV_OFFERED,
GCEV_ACCEPTED,
GCEV_ANSWERED

IVR Events 1

USREV_NEWCALL,

To add a party to a new conference, we receive:

DMEV_CONNECT x2,
CNF_ADD_PARTY

When we get a disconnect, we remove the parties, so we receive:

GCEV_DISCONNECTED,
GCEV_DROPCALL,
GCEV_RELEASECALL,
DMEV_DISCONNECT x2
CNF_REMOVE_PARTY

This indicates that for the simplest call we have 13 events.

To this we need to add the switching events. In this case, an event triggers the switch (simulating the digit event). Assume that the announce feature is available on where to route to a voice device and then do the play. Thus, for each switch we receive:

USREV_SWITCH,
CMF_REMOVE_PARTY,
DMEV_DISCONNECT,
GC_LISTEN,
TDX_PLAY,
GC_UNLISTEN,
DMEV_CONNECT
CNF_ADD_PARTY

This indicates that we receive 8 events.

Next, take a look at the rate at which the participants switch conferences. Assume that each participant spends an average of 30 seconds in each conference. That indicates that the user is going to generate 8 events every 30 seconds.

Next, assume that the average user spends 10 mins (600 secs) in the server.

With the above information, we can assume that each user will generate 173 events on average: 13 for the basic setup, plus 8 every 30 secs for 10 mins:

13 + (600/30)*8 = 173

So, if each call generates 173 events, we can figure out the rate per second. The call lasts for 600 seconds, so each call will generate 173/600, or, on average, .288 events per second.

Note: As the call duration increases, the call setup events become less of an impact and may be able to be ignored.

Then, we look at average number of callers. If we have 300 calls, our users will be generating on average 300 * 0.288 or 86.4 events per second, or one event every 11.57 ms (1000/86.4).

If our process event procedure is ever taking longer than 11.57 ms, some performance may be impacted for a period of time until the SRL can process out the event queue. Thus, you want the SRL event queue to be empty as much as possible.

The following steps can handle this:

1. Spend minimal time inside the SRL event processing procedures. If you find you are spending more time than is needed, then offload processing of those tasks to worker threads to be completed. You will note that in this application, there was a performance point where the amount of time inside the ProcessEvt function was tracked (Process Event CB). This should be zero. Use the –cb option in the demo to view the relative impact that spending additional time has on performance.
2. Eliminate superfluous events. For example, for times when routing of CNF devices to parties can be maintained, there is no need for DISCONNECT and RECONNECTS, which can reduce the number of generated events.
3. Build logic into your application that can detect the burst, and handle and scale accordingly. One way to do this is by placing a Heartbeat event onto the event queue with sr_putevt and determining how long it takes to come into the process event routine. If this time is longer than some threshold value, you may have to limit access to the server or prevent a new request from starting until the SRL empties and performance is restored.
4. Make sure the application has a dedicated event processing thread. If it does not, then you have to calculate the percentage of time to process events and adjust your callback time accordingly.

Network Tuning

As seen in the test runs in this application note, if you want to deploy a high density IP solution, tune the IP network. Documents and guidelines are available for architecting a network to make it conducive to VoIP; thus, those specifics are not covered in this application note.

However, on a high level, it is important that you have your network correctly divided into subnet (and utilize masking correctly). Allocate enough bandwidth for VoIP traffic (that is, gigabit is preferred for any larger scale deployment) and ensure that the infrastructure is capable of handling large scale VoIP traffic.

As seen from the above testing, failure to do this will result in some impact to the voice quality performance.

For More Information

A Zip file containing the application code can be downloaded at http://www.dialogic.com/goto/?10831

Conferencing API Programming Guide — http://www.dialogic.com/manuals/docs/conferencing_programming_v1.pdf

Conferencing API Library Reference — http://www.dialogic.com/manuals/docs/conferencing_api_v1.pdf

Overcoming Barriers to High-Quality Voice over IP Deployments — http://www.dialogic.com/goto/?8539

Dialogic

To learn more, visit our site on the World Wide Web at **http://www.dialogic.com.**

Dialogic Corporation
9800 Cavendish Blvd., 5th floor
Montreal, Quebec
CANADA H4M 2V9

11/07 10787-01