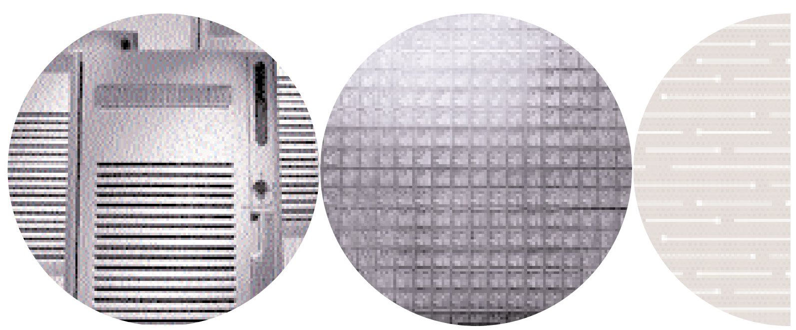
Reference Design

Host Media Processing for Windows Operating Systems Reference Design Guide

Intel in Communications



Contents

Figures

Tables

About This Publication

The following topics provide information about this publication:

- Purpose
- Intended Audience
- How to Use This Publication
- Related Information

Purpose

This publication provides a general overview of host media processing (with a business focus), describes in more detail the Intel® NetStructure™ Host Media Processing software solution configuration, describes how to build an interactive voice response (IVR) solution, provides guidelines for writing applications, describes how the solution configuration was tested, and provides performance data.

Intended Audience

This publication is written for the following audience:

- Distributors
- System Integrators
- Toolkit Developers
- Value Added Resellers (VARs)
- Original Equipment Manufacturers (OEMs)
- Telephony Equipment Manufacturers (TEMs)

How to Use This Publication

The information in this guide is organized as follows:

- Chapter 1, "Host Media Processing Software Description" provides a high-level description of the Inte® NetStructure™ Host Media Processing (HMP) software, discusses market segment opportunities and environment, provides general cost information, describes host media processor users, and summarizes the business and market segment.
- Chapter 2, "Host Media Processing Configuration Description" describes the IVR configuration, lists HMP 1.0 features, describes various IVR application scenarios, and shows the architecture of the IVR reference application.

- Chapter 3, "Installing the Host Media Processing Software" provides procedures for installing the Inte® NetStructure™ Host Media Processing (HMP) Software. This chapter contains excerpts from the *Intel® NetStructure™ Host Media Processing Software Release 1.0 Installation and Configuration Guide*, and you can refer to that document for additional details.
- Chapter 4, "HMP Licensing Procedure" describes the procedures for obtaining an evaluation run-time license and purchasing a permanent HMP license. This chapter contains excerpts from the *Intel® NetStructure™ Host Media Processing Software Release 1.0 Installation and Configuration Guide*, and you can refer to that document for additional details.
- Chapter 5, "Configuring the Host Media Processing Software" provides information about configuring the system using the Intel® Dialogic® Configuration Manager (DCM).
- Chapter 6, "Understanding the IVR Reference Application" provides information about the IVR reference application that includes descriptions of a basic state machine and the simple device concept.
- Chapter 7, "Solution Configuration Testing and Performance" describes the test methodology and test scenarios, and provides performance data regarding CPU utilization, DTMF digit detection, and busy hour call attempts.
- Chapter 8, "Application Design Alternatives for Your Solution" provides information about consulting services from Intel.

Related Information

The documents listed in this section are available on the electronic bookshelf that is provided with the Intel® NetStructure™ Host Media Processing Software. The bookshelf is also available at this URL:

http://productinfo.py.intel.com/Bookshelf/HMP1.0_Windows/index.html

The information in Chapter 3, "Installing the Host Media Processing Software", Chapter 4, "HMP Licensing Procedure" and Chapter 5, "Configuring the Host Media Processing Software" was excerpted from the installation and configuration guide listed below. You can refer to this document for additional information about installation and configuration.

- *Intel® NetStructure™ Host Media Processing Software Release 1.0 for Windows Installation and Configuration Guide*
- *SNMP Agent Software for Host Media Processing Software Release 1.0 for Windows Administration Guide*
- *IP Media Library API for Linux and Windows Library Reference*
- *IP Media Library API for Linux and Windows Programming Guide*
- *Voice Software Reference: Features Guide for Windows*
- *Voice Software Reference: Programmer's Guide for Windows*
- *Voice Software Reference: Standard Runtime Library for Windows*
- *Dialogic Audio Conferencing Software Reference fo Windows*
- *Global Call Application Developer's Guide for Linux and Windows*

- *Global Call API Software Reference for Linux and Windows*
- *Global Call IP Technology User's Guide for Linux and Windows*
- *Compatibility Guide for the Dialogic R4 API on DM3 Products for Linux and Windows*
- *DM3 Diagnostic Utilities Reference Guide*

Additional information about the products used in the solution configuration is available online at this URL: *http://www.intel.com/design/network/products/telecom/index.htm*.

Host Media Processing Software Description 1

This chapter describes the Intel® NetStructure™ Host Media Processing (HMP) software product, discusses market segment opportunities and environment, provides general cost information, describes HMP users, and summarizes the business and market segment. Major topics include:

1.1 Overview

The following sections provice an overview of the HMP product:

- Product Description
- Host Media Processing Applications
- HMP Software Reference Architecture

1.1.1 Product Description

Intel® NetStructure™ Host Media Processing (HMP) software performs media processing tasks on general-purpose servers based on Intel® architecture without the need for specialized hardware. The software provides media services that can be used to build flexible, scalable, and cost-effective next-generation IP media servers.

HMP software is an Intel® communications building block technology. When installed on a system, the software looks like an Intel® Dialogic® board with DM3 architecture to the customer application, but all media processing takes place on the host processor. To help customers accelerate their time to market and migrate their existing applications to IP, the software also supports two direct application programming interfaces (APIs): R4 for media processing and Global Call for call control.

HMP software makes use of a built-in network interface card (NIC) to provide IP connectivity. It supports the industry-standard H.323 protocol for call control, and RTP/RTSP for media streaming over IP in G.711 format. To aid the quality of media streaming over the network, the software supports frame sizes of 10, 20, and 30 ms, as well as additional features such as QoS threshold alarms.

HMP software is implemented as a Windows* operating system kernel-mode driver that runs at real-time priority. The software is optimized to run on the Intel® Pentium® III, Pentium® 4, and Xeon™ processors. Since HMP software is implemented as a software-only product, it can be installed and upgraded like any other software. The software is licensed using an industry-standard model in which the MAC address is used to node-lock the software to a specific computer. To enable customers to choose combinations of media processing, HMP software is offered in a number of licensing models.

The host-based resources provided by the HMP software include:

- Player resources
- Recorder resources
- Tone detection and generation resources
- IP telephony media resources
- Conferencing resources

After the HMP software has been installed, the host-based "soft-DM3 board" emulates an IP resource board, and configuration files can be "downloaded" in the same way as to a physical board.

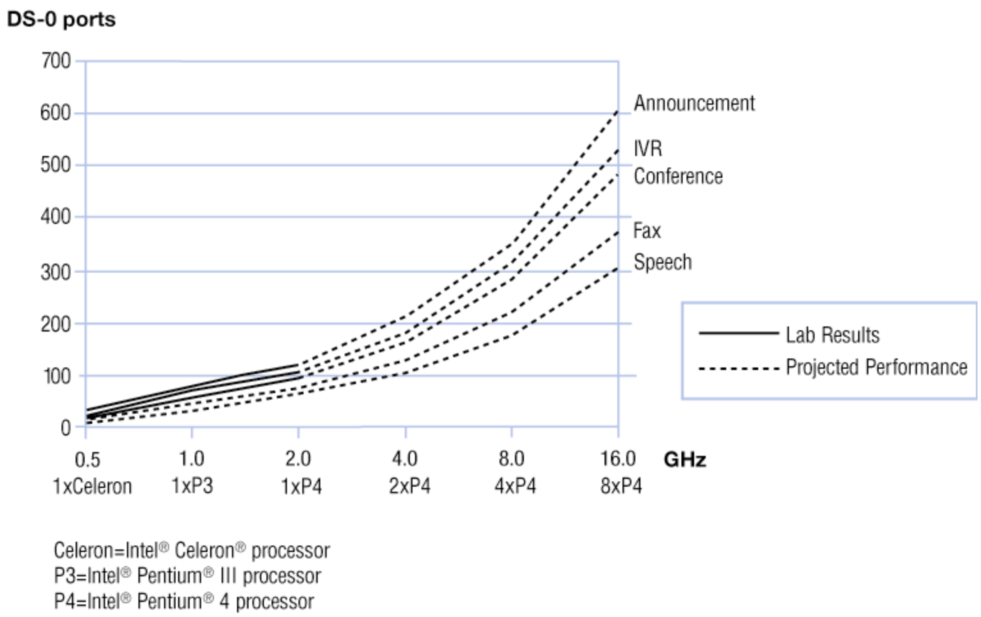
1.1.2 Host Media Processing Applications

The introduction of digital signal processing capabilities on the Pentium processor has made it an extremely cost-effective technology for deploying host media processing functions. According to preliminary estimates, Intel anticipates that HMP software will enable a single 1GHz Pentium processor to replace two 150 MHz DSPs. From this baseline, Moore's law improvements to the Pentium processor will enable HMP software at densities of DS-3 and even OC-3. Figure 1 shows lab results and the projected performance of a number of major communication media processing functions:

1. **Announcements** - Streaming µlaw or Alaw audio data from a file on a hard disk and converting into a telecom audio stream through a linear player resource
2. **Interactive Voice Response (IVR)** - Combining announcements with a Dual Tone Multi-Frequency (DTMF) or touchtone signal detector algorithm. This function includes barge-in, a feature that stops the announcement when a DTMF digit is detected or when speech is detected. This Reference Design Guide will focus on implementing an IVR application using HMP software.
3. **Conferencing** - Bridging voice streams in a three-way conference
4. **Speech** - Performing a range of speech technology functions, including TTS, speech menus, and large vocabularies (included only for comparison)

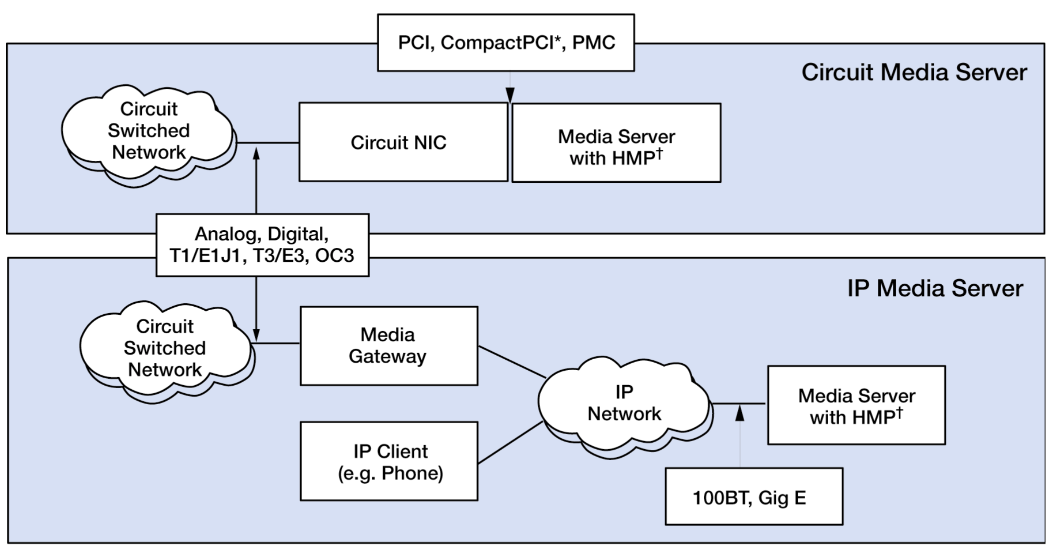
The HMP software program at Intel has implemented, optimized, and tested announcements, interactive voice response (IVR), and conferencing algorithms on platforms with Intel® Celeron® or Pentium® processors ranging from 566 MHz to 2 GHz. The solid lines in Figure 1 represent the lab results. Intel is targeting an implementation of the algorithms that scales the technology on higher speed Intel® Pentium® processors, Intel® Pentium® processors on two-way, four-way, and ultimately eight-way servers, Intel® Xeon™ processors, and Intel® Itanium® processors. The dotted lines in Figure 1 project Intel's target results for some of these future implementations.

Figure 1. Lab Results and Projected Performance



The modular next-generation network combines voice and data in a single packet infrastructure. As a result, two main architectures are currently feasible for deploying media services: a direct circuit-based network interface and a packet-based IP network interface. These two architectures are shown in Figure 2. HMP software can provide media processing in both.

Figure 2. Media Server Deployment Architecture



† Intel® Netstructure™ Host Media Processing software

Packet-based IP network deployments only require Ethernet network interfaces. New, and more recently, traditional service providers are moving to standard 100BaseT and Gigabit Ethernet as the primary interface for both Internet backbone services and telecom voice connections. Service providers are deploying media gateways to convert the circuit network voice stream to an Ethernet-based infrastructure using voice over IP (VoIP). The ubiquity of Ethernet has necessitated equipping standard Web computing platforms, and now desktop and laptop computers, with Ethernet interfaces. As a result, an Ethernet telecom media server can be deployed on a standard Web computing platform with HMP software alone and without any additional voice processing hardware. These Ethernet networks allow Ethernet-based telecom media servers to provide voice media processing for both legacy circuit clients and IP-based clients.

The same type of Web server platform described above is currently being used for call processing functions in softswitch applications. Telecom media processing services can easily be added to these networks with HMP software running on the same server platform as the softswitch. This new type of deployment would replace the specialized media processing platforms currently in use. The economics of using homogeneous hardware architecture are very compelling. Total cost of ownership is significantly reduced for both network and enterprise service providers because the costs of sparing, training, and the integration of management systems are so much lower. Such a deployment also speeds time-to-market and accelerates the innovation cycle by allowing new features to be deployed through a software upgrade alone.

Deploying a media server with a connection to the traditional circuit switch network or an Asynchronous Transfer Mode (ATM) network requires the addition of a circuit network interface card (NIC). A circuit NIC provides either the circuit switched line interfaces (analog [loop start, ground start] or digital [ISDN BRI/BRA]), or the circuit switched high capacity trunk interfaces (T1/E1/J1, E3/T3, OC3). A typical circuit NIC includes the following basic functions:

- A line interface unit (LIU) to terminate the copper wires or fiber
- A framer for digital interfaces to terminate the layer 1 protocols or a COder DECoder (CODEC) for analog interfaces for conversion from analog voice signals to digital signals
- A standard peripheral component interface (PCI) streaming interface to the computing platform in PCI AT , CompactPCI*, or PCI mezzanine card (PMC) form factors

For high capacity trunk interfaces, the circuit NIC could also include specific components to optimize time division multiplexing (TDM) channelization data and perform high level data link control (HDLC) controller function processing for ISDN and Signaling System 7 (SS7) signaling

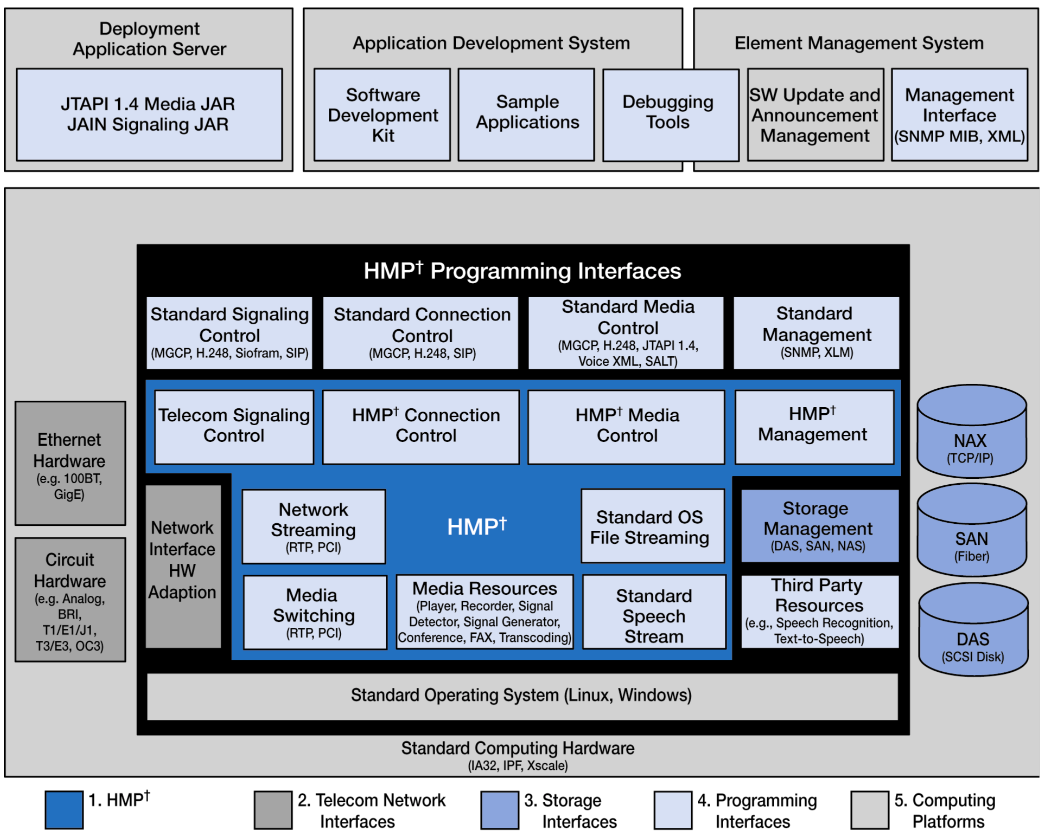
The circuit NIC streams data to the computing server platform with very low latency (typically less than 8 milliseconds). The computing server platform then performs the media processing and, if applicable, call control signaling processing as well.

In support of the growth of data networking, high density (T1/E1/J1, T3/E3, OC3, ATM) NICs are becoming more common in the market segment. Simple circuit NICs enable low cost plug-ins into standard routers as a wide area network (WAN) interface. They also allow computing platforms to perform routing functions. As the design of circuit NICs becomes standardized, their evolution will be similar to Ethernet NICS in their market segment, but more limited. This evolution should drive down the cost and move circuit NICs towards the same "plug and play" compatibility that is now available with Ethernet NICs. Intel's standard "plug and play" interface for processing circuit NIC data is described in greater detail in the next section.

1.1.3 HMP Software Reference Architecture

Intel has developed the HMP software reference architecture, shown in the block diagram of Figure 3, as a blueprint for the components in a telecom media server solution. This architecture defines the requirements for the interfaces and functionality to support both circuit-based and packet-based media servers. Please note that this is a conceptual diagram only, and the actual HMP software products offered by Intel do not necessarily include all the components shown on the diagram.

Figure 3. HMP Software Reference Architecture Block Diagram



† Intel® Netstructure™ Host Media Processing software

The reference architecture has five major functions:

1. HMP software
2. Telecom network interfaces
3. Voice and data storage interfaces
4. HMP software programming interfaces
5. Standard computing platforms

Each of these functions is described in detail in the following sections.

1.1.3.1 HMP Software

HMP software supports two major functions: voice media processing algorithms and software voice switching.

Voice Media Processing Algorithms

The voice media processing algorithms support the functionality for voice processing resources: announcements, IVR, conferencing, and transcoding. However, developing algorithms is insufficient. To create a robust voice media-processing framework, two other areas require attention:

- Tuning the algorithms for network deployment
- Optimizing the algorithms for the platform

Intel's algorithms are derived from Intel® Dialogic® and Intel® NetStructure™ products. These algorithms have been tuned for use on telecom networks through millions of ports deployed in more than 100 countries over the last twenty years. Intel is porting this technology to the Intel® Celeron®, Pentium®, Itanium® and XScale™ processor families.

Intel optimizes these core algorithms by recoding the implementations in C to most effectively utilize the MMX™ and SSE technologies in Pentium® and Itanium® processors. Optimization can produce up to a tenfold increase in the performance of the algorithm over the initial C code version. For example, by rewriting the basic C algorithm to utilize SSE instructions, Intel expects to increase the efficiency of the G.729a algorithm from 200 MHz to less than 20 MHz per instance of the resource.

Intel is also working with speech technology vendors to reduce costs and simplify the deployment of speech applications. Applications based on speech recognition and text-to-speech (TTS) technology (e.g., speech IVR, voice portal) provide compelling benefits in terms of operational efficiency and competitive differentiation. An Intel technology called continuous speech processing is an example. This technology optimizes voice activity detection (VAD) algorithms to meet the stringent performance requirements of speech recognition. VAD only streams data to the speech engine when actual human speech is detected, thus optimizing the number of millions of instructions per second (MIPS) that a Pentium processor requires to support speech recognition. According to preliminary estimates, Intel anticipates that a 1 GHz Pentium III processor can support over 100 channels of streaming to a speech engine. The result is an increase in densities and a lower cost when deploying speech-enabled solutions.

Software Voice Switching

Intel's optimization efforts extend beyond core media processing algorithms to include the framework's media switching core. The core supports switching voice data to and from resources, to and from files, and to and from third-party resources (e.g., speech recognition and TTS).

Media switching must move data to and from the network interfaces with an extremely low latency. Total end-to-end latency must be less than 200 ms for real-time-sensitive services such as conferencing so that a human user does not perceive a time delay. The HMP software framework will support conferencing with less than 50 ms of latency to ensure an end-to-end latency of below 100 ms.

Another optimization example is VoIP real-time transfer protocol (RTP) processing on an Ethernet media server. Intel is optimizing its Ethernet driver implementation to reduce the overhead of RTP-based streaming and switching in the platform.

1.1.3.2 Telecom Network Interfaces

Network interfaces provide the connection to the circuit network or client device (e.g., analog or digital phone) as defined in the Deployment Environments section above. The HMP software reference architecture defines two types of network interfaces:

1. Ethernet network interface card (Ethernet NIC)
2. Circuit network interface card (circuit NIC)

These interfaces stream data to the computing platform through a standard PCI driver that is part of the computing server operating system (OS).

Ethernet Interface

The Ethernet NIC streams voice data in and out of the computing platform based on the standard IETF RTP. The media-processing framework unbundles the RTP data and streams the media content to the media processing resources under the control of the application. Data originating from an announcement storage or media resource is likewise streamed from the media-processing framework to the Ethernet NIC. (Most currently available computing servers include dual Ethernet NICs.) As carriers and enterprises move to a homogenous Ethernet architecture for both voice and data, voice-enabled computing servers can plug into the networks with no additional hardware. This converged network will make implementing solutions faster and less expensive.

Circuit Interface

The circuit NIC streams voice data in and out of the computing platform through the PCI bus in buffered TDM form. As with the Ethernet interface, the media-processing framework unbundles the TDM data and passes it to the media-processing framework. Trunk circuit interfaces typically require the interface to perform echo cancellation before engaging a specific resource like IVR, conferencing, messaging, or speech recognition. Intel expects that a 1 GHz Pentium III processor will support over 150 channels of echo cancellation.

The circuit media interface also streams digital signaling protocols (ISDN or SS7). These protocols require HDLC or Message Transfer Part 1 (MTP1) processing and level 2 Link Access Protocol-D/Message Transfer Part 2 (LAP-D/MTP2) processing fix order/mixing layers. Terminating these protocols is done through one of the following architectures:

- Dedicated hardware on the circuit media interface board for level 1 processing
- Level 1 software processing on the computing platform

Utilizing dedicated hardware on the circuit NIC has the advantage of offloading the overhead of the level 1 processing from the computing platform. Intel expects to support both architectures. In both cases, the signal protocol is terminated at level 2, and the level 3 packets are sent or received from the application through either the signal control interface or a standard protocol such as

Simple Computer Telephony Protocol ISDN User Adaptation layer (SCTP IUA) for ISDN or Simple Computer Telephony MTP3 User Adaptation layer (SCTP MxUA) for SS7.

A standard circuit NIC interface will allow service providers to choose between multiple circuit NIC interface types (e.g., analog, BRI, T1/E1/J1, DS-3, etc.) and ideally supply vendors with "plug and play" compatibility with the HMP software architecture.

1.1.3.3 Voice and Data Storage Interfaces

Most telecom media processing applications require the ability to play announcements. For example, network call centers for automated 800 services can require storage of over 100,000 announcement files that allow for multiple language support and multiple service support. Files are also the basis of storing voice mail and fax mail in messaging applications. To facilitate the fulfillment of these requirements, HMP software supports the streaming of data from files and the storage of data to files on standard computing platform disks or flash.

HMP software streams data from a file using standard OS file system primitives. This enables the operator to choose the type of storage used.

1.1.3.4 HMP Software Programming Interfaces

To allow integration with various industry architectures and mapping across standard industry interfaces, programming interfaces are divided into four areas:

- Call signaling or call control
- Connection control
- Media control
- Management or administration

The programming interfaces are designed to provide two levels of integration across these four functional domains. HMP software provides direct C-language programming interfaces for call control (also referred to as Global Call) and media processing (also referred to as R4 API). These interfaces are backward compatible with those offered with Intel® NetStructure™ DM3 board-level commnuications products, enabling customers to use the same applications for these products and HMP software.

In addition, industry-standard programming interfaces can be provided that enable Plug and Play interfaces to solutions such as softswitches and application servers.

Call Signaling or Call Control

Call signaling or call control provides the services that interact with the network to set up a voice session between two voice endpoints. Using one of the five major voice network call signaling protocols typically does this: inband (e.g., R1 or R2), ISDN, SS7, H.323, or session initiation protocol (SIP).

Call control is an optional part of HMP software. It is defined in the media server reference architecture to the extent necessary to specify a reference to media processing resources and enable the streaming of the call signaling data associated with the media stream to the application level call processing functions (e.g., softswitch). In addition, it is useful to define call control to support an all-in-one-solution.. The current release of the HMP software includes the H.323 call control stack from Radvision, with support of the Global Call direct C-language programming interface. HMP software also provides a mechanism that enables customers to integrate any 3rd-party call control stack with HMP software.

In general, the architecture assumes that an application will interact with call control outside of HMP software and will define the association to set up a voice connection to the media processing resource through connection control. This offers the OEM and service provider optimal flexibility to integrate any major call control architecture as required by an application's specific features.

Connection Control

Connection control gives the application control when linking the voice stream (either circuit digital signal level 0 [DS-0] or IP RTP session) to a media resource by controlling the HMP-software-switching configuration. The connection control interface also allows the application to create a connection between two voice streams (e.g., two individuals in a call). HMP software provides a connection control interface based on the DM3 architecture model. This model allows granular control when setting up RTP sessions, circuit NIC connections, and resource connections in both uni- and bi-directional modes.

With advantages very similar to the those of IP, a software-based architecture can move data by passing pointers to shared memory segments. This flexibility simplifies the development of advanced applications for conferencing, call center, and other services by enabling a variety of connection architectures between resources. For example, a configuration of 100 DS-0s could include 100 ports for IVR and only 20 ports for conferencing. The flexible switching in the software solution allows the conferencing ports to be brought in dynamically and trade the MIPs of a Pentium processor with the IVR resources. In a hardware implementation, conferencing is typically dedicated to either all the resources on a board or a percentage of them. Resources cannot be traded between applications easily.

Also, by using software-based switching and an Ethernet fabric, architecture can be scaled for a large switching implementation using standard computing hardware. For example, instead of configuring conferencing as a shared resource with IVR on a server, conferencing can be configured as a separate resource in the network. Ten different IVR servers could connect to the conferencing server over IP when a conferencing resource is required.

Media Control

Media control is central to the software that provids the programming interface for all of the HMP software resources and algorithms. The direct media control interfaces are fully compatible with DM3 series application programming interfaces (APIs) for media written in C. Thus, a rich set of applications developed for Intel® Dialogic® products during the past twenty years can now benefit from a software-only architecture when HMP software is used.

The direct media control interfaces provide the granular level of algorithmic control required for highly tuned applications (e.g., engage a specific echo canceller). Included is the control of required algorithmic thresholds and configuration, and input and output (e.g., DTMF digits).

In the area of standard interfaces, five telecom media control-programming interfaces have been defined by the industry: Media Gateway Control Protocol (MGCP) Audio Server, H.248 Annex M (H.248.9), Java* Telephony API (JTAPI) 1.4 Media, VXML, and SALT.

MGCP and H.248 are connection control protocols enhanced to support media services that could be deployed on a media gateway or media server. In the case of next-generation softswitches, the enhancements allow the use of one protocol to control all network elements: media gateways or media servers. MGCP and H.248 are focused on the core media functions required by the legacy telecom network (e.g., network announcements, operator services, and three-way calling). MGCP interfaces provide a natural mechanism to explicitly identify coders for transcoding from compressed speech formats to uncompressed formats.

JTAPI 1.4 Media, VXML, and SALT are targeted to enable advanced telecom services. JTAPI 1.4 Media (ECTF S.410) is a Java interface specifically defined to support both fundamental media services and advanced speech services. It has not yet gained the same level of acceptance as VXML.

VXML and SALT have been defined specifically to support the integration of speech recognition capabilities with Web-based application environments. VXML has gained a large following in the Web services programming model for voice portal and ultimately speech-driven call center applications. SALT extends VXML concepts to include support for standard Web scripting languages (e.g., JavaScript) and multimode applications (e.g., an Internet-enabled phone).

The modular architecture and powerful yet highly usable direct programming interfaces of HMP software enable easy integration with any of the standard programming interfaces described above.

Management or Administration

The comprehensive and standard management frameworks provided by computing platforms as part of their operating systems provide significant interoperability with both carrier and enterprise management systems. Platforms typically support both Simple Network Management Protocol (SNMP) and eXtensible Markup Language (XML) management for processor performance, and Ethernet interfaces and disks. In addition, Intel computing platforms also support the IP management interface (IPMI) for monitoring the platform and providing the robust management interface required for lights-out server operation.

HMP software provides support for three critical management areas necessary for reliable operation and effective engineering: performance, fault, and configuration. The management information base (MIB) for telecom network interfaces, storage, and computing hardware are based on the standard industry capabilities included as part of those components.

Intel is also implementing standard MIB bindings for both the Linux* and Windows* operating systems. Intel is focusing on an SNMP binding for performance and fault management and XML over Simple Object Access Protocol (SOAP) for configuration management. In addition, to support management over the Internet, SNMP Version 3 is being targeted as critical to enhanced security capabilities.

1.2 HMP Business Drivers, Market Segment Opportunities, and Cost

HMP software, also referred to as "software-only media processing", is a technology that enables you to use general-purpose processors to perform media processing tasks. This technology brings numerous advantages to the telecommunications solutions developers, most importantly cost savings. So, recently there has been a spate of new telecom products built using this technology at different levels of integration, from building block components to complete turnkey solutions, and everything in between. This technology is now drawing attention of market research analysts, References 9 and10.

Since HMP software fundamentally changes the way media processing in telecommunications is done, it is often referred to as revolutionary or disruptive. There are two main drivers behind this technology:

- The newest, most powerful Intel® processors enable a standard computing platform to cost-effectively perform the media processing functions done previously only by the special-purpose processors.
- Migration to Voice Over IP (VoIP) enables you to use an Ethernet card (which is embedded into virtually every computer built today) as a network interface for connecting to the telecommunications networks.

Combining these two factors enables developers to build software-only telecommunications media processing servers using standard high-volume (SHV) servers, without any specialized hardware components (currently, application developers need to buy boards and a computing platform to begin developing new telecom applications).

With HMP software, developers can start work on new telecom applications after a simple software install. Because of this, costs and deployment barriers for media processing servers decrease substantially.

Specifically, the costs are reduced in the following areas:

- Initial investment - customers pay only for the media resources they need
- Development - more cost-effective development/test systems
- Deployment - no need to keep inventory, also less shipping, import/export fees, no physical install and reduced regulatory approvals
- Post-deployment - sparing costs reduced or eliminated, easier field upgrade (could be done remotely), improved overall system availability (less hardware components)
- Upgrades - flexibility in adding media processing

Reduced costs, along with emerging Internet-telecom integration, promises to spawn a new wave of innovative, cost-effective telecom applications

Intel has created the HMP software for customers to take advantage of the cost savings offered by HMP technology. This product provides features similar to those offered on Intel® NetStructure™ boards, but implemented completely in the software on the Intel® architecture processors. It enables customers to build IP media servers without using any specialized hardware, and provides the same APIs as the Intel® NetStructure™ boards.

The following sections include discuss the economics and business opportunities behind the main HMP technology drivers.

1.2.1 Moore's Law Makes HMP Possible

Intel first introduced basic digital signal processing functions for its processors with Intel® MMX™ technology on the Intel® Pentium® processor. MMX™ technology began a major enhancement to the Intel architecture that was designed to accelerate multimedia and communications software. This technology essentially embedded basic DSP capabilities in the Pentium processor and included new data types and 57 new instructions to accelerate calculations common in audio, 2D and 3D graphics, video, speech synthesis and recognition, and data communications algorithms. As the number of transistors on a chip has grown, Intel has continued to enhance the digital signal processing ability of the Pentium® processor.

The capability that was created with MMX technology has been expanded with Streaming SIMD Extensions (SSE). Streaming SIMD (single instruction-stream multiple data-stream) extensions are instructions that reduce the overall number of instructions required to execute a particular program task. As a result, they can boost performance by accelerating a broad range of applications, including video, speech, and image, photo processing, encryption, financial, engineering, and scientific applications. Intel® NetBurst™ microarchitecture adds 144 new SSE instructions, which are known as SSE2 and are available on the Intel® Pentium® III and Celeron® processors.

Today, Intel® processors perform floating point operations, matrix operations, fast Fourier transform, finite Impulse response and multi-threading. In addition, Intel provides Intel® Integrated Performance Primitives to optimize digital signal processing.

For optimized I/O required for signal processing, Intel processors offer wider data buses and faster memory buses. For multiple-access memory architectures, they offer large on-chip caches with multiple operands per fetch.

With Moore's Law expected to work for at least another decade, according to a recent C-NET article, HMP technology has a significant growth potential.

1.2.2 Economics of VoIP

According to market research, recent economic conditions have slowed down telecom spending in both Enterprise and Service Provider market segments, which negatively affected investment in new telecommunications products.

The same factor, however, facilitates migration to Voice over IP (VoIP, also referred to as Voice Over Packet or VoP) as more cost-effective technology in the long term, as indicated in Reference 9 and a number of other sources.

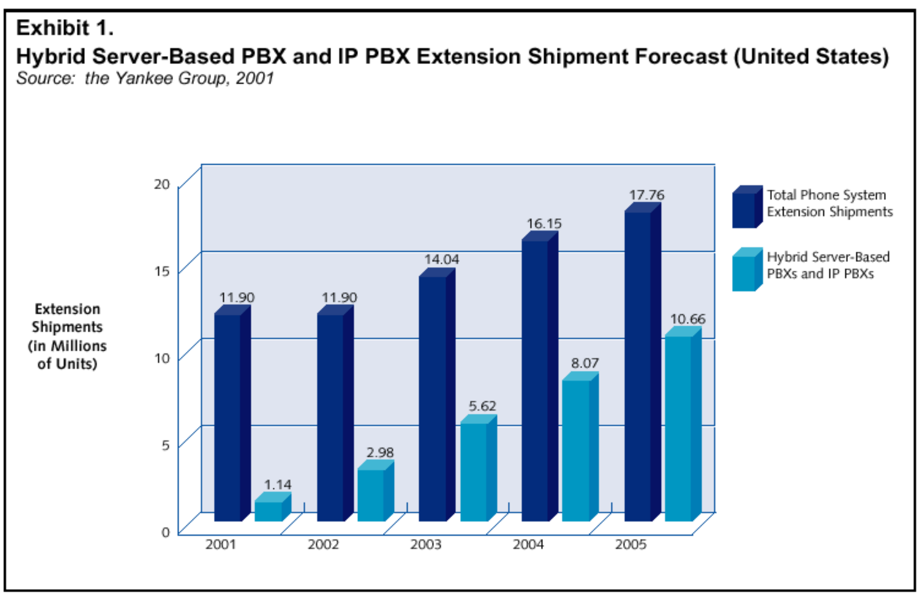
Currently, PSTN still represents the majority of traffic in all market segments, as shown below. However, migration to IP is still taking place for both Enterprise and Service Providers, and expected to accelerate in the next 1-2 years, with a number of researches calling 2004 an inflection point where we will see a major growth of VoIP deployments. For instance, Reference 6 projects 50% CAGR for IP telephony until at least 2006. The following sections describe the VoIP trends in the Enterprise and Service Provider segments in more detail.

1.2.2.1 IP Penetration for Enterprise

As stated in Reference 2, “The combined yearly IP shipments are expected to surpass the 50 percent penetration level (in terms of annual total shipments) in the global CPE market by 2006 (although in some regions such as North America, the cross-over point is forecast to occur earlier in 2005). This is a good indicator of the potential of the technology, given that in 2001, total IP-enabled and IP-centric PBX shipments constituted only 5.12 percent of the total premise switch equipment lines globally sold”. In the same source, the entire Enterprise IP telephony TAM is estimated at $1.2B in 2003.

These numbers roughly correspond to the ones shown in Figure 4.

Figure 4. Hybrid Server-Based PBX and IP PBX Extension Shipment Forecast (United States)



Reference 2 also shows the forecast for the number of lines for IP-enabled (legacy PBXs with added IP connectivity), Figure 5, and IP-centric (next-generation converged and pure IP PBXs), Figure 6.

Figure 5. IP-Enabled PBX Market Segment: IP-Enabled Lines and Revenue Forecasts

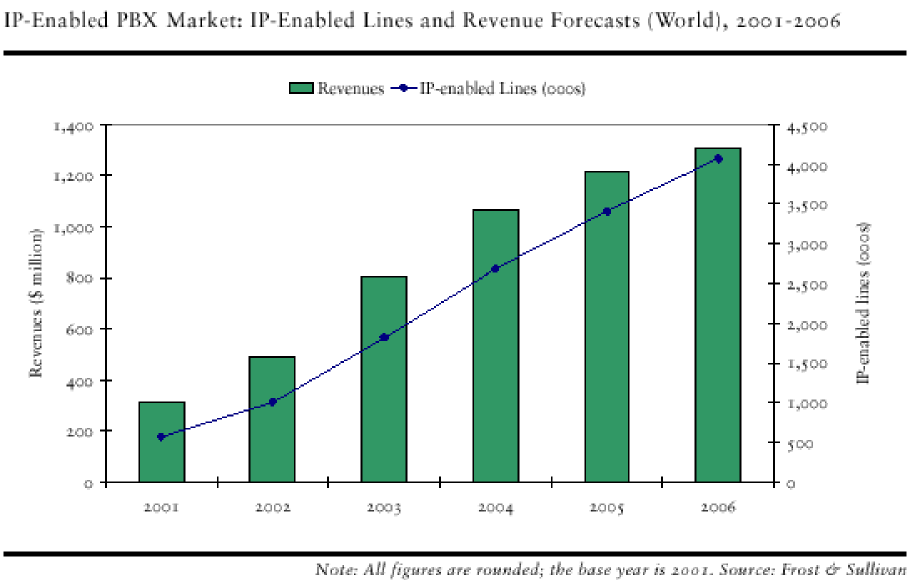
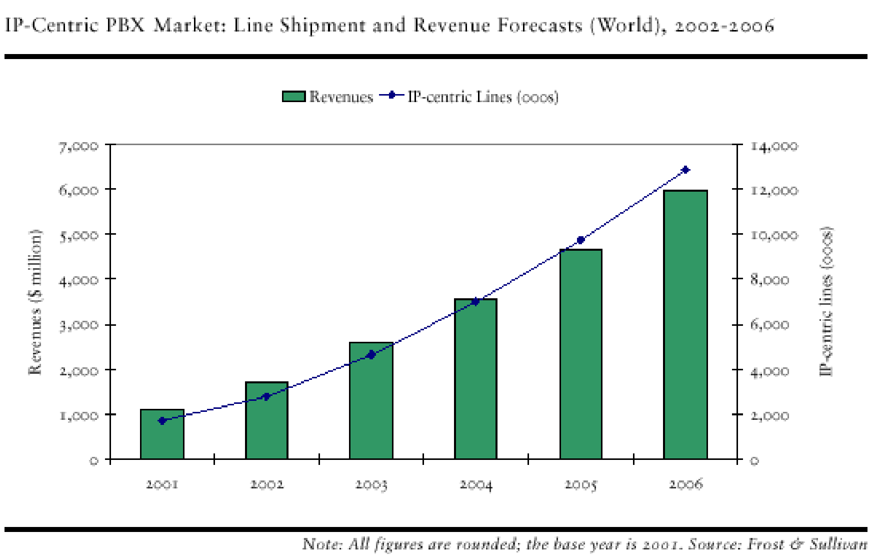


Figure 6. IP-Centric PBX Market Segment: Line Shipment and Revenue Forecasts



1.2.2.2 IP Penetration for Service Providers

Currently IP adoption by Service Providers is at a relatively small level. According to References 5 and 7, in 2001 VoIP represented approximately 5% of the traffic worldwide. A number of reports indicate that it will grow substantially within the next few years.

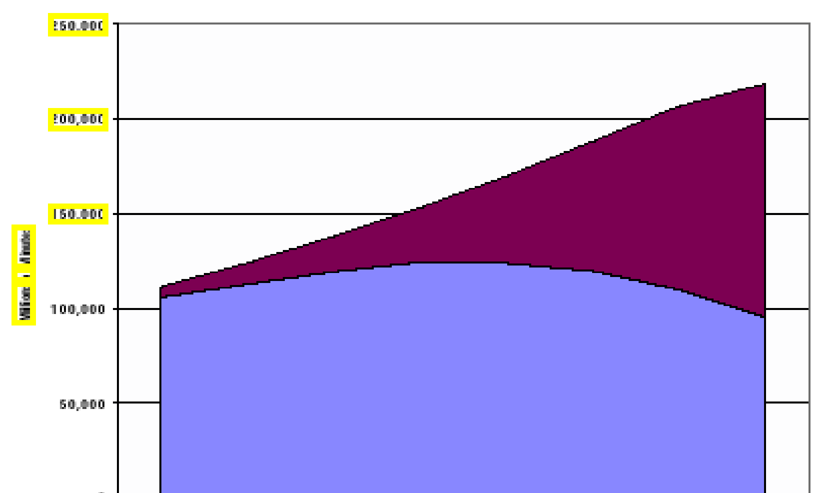
Key market segment drivers for the Service Provider market segment are as follows, according to Reference 6:

1. Declining prices of telecom services lead carriers to seek new revenue sources
2. SIP's emergence on the radar screen of carriers
3. Competitive pressures facing incumbents
4. Deregulation is opening up new markets for VoIP
5. Global growth trends for telephony
6. RBOCs getting 271 relief for long distance
7. Network optimization and consolidation
8. Keeping shareholders happy

Figure 7 shows the trends in international voice over packet (VoP) traffic and is consistent with the numbers shown in Figure 8.

Figure 7. International TDM and VoP Traffic, 2000 - 2007

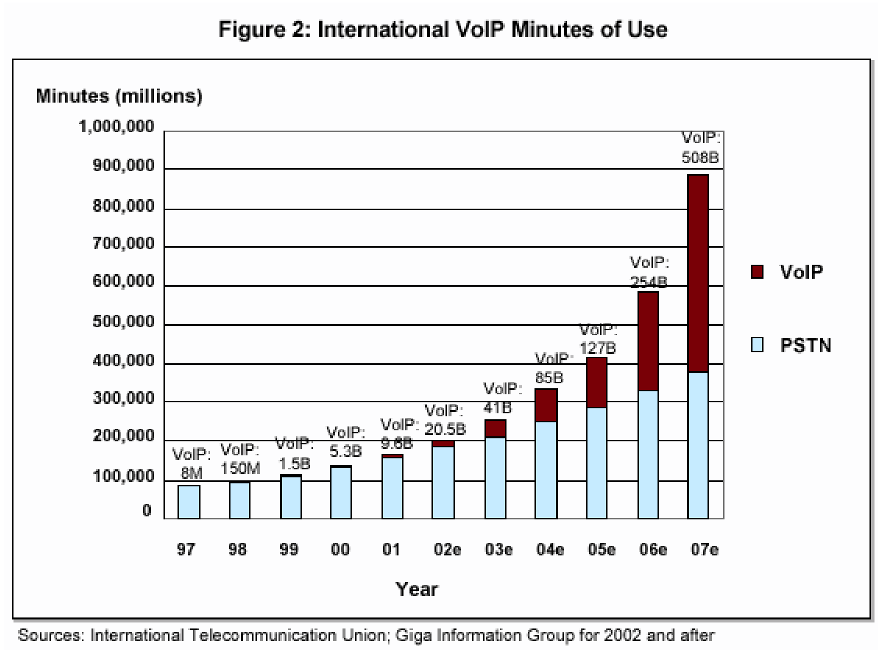
Chart 1: International TDM and VoP Traffic— 2000-2007



	2000	2001	2002	2003	2004	2005	2006	2007
VoP	5,544	11,021	18,502	28,678	45,881	68,472	96,926	123,311
TDM	105,766	112,779	119,468	124,446	123,871	119,650	109,918	95,296

Source: Probe Research, Inc.

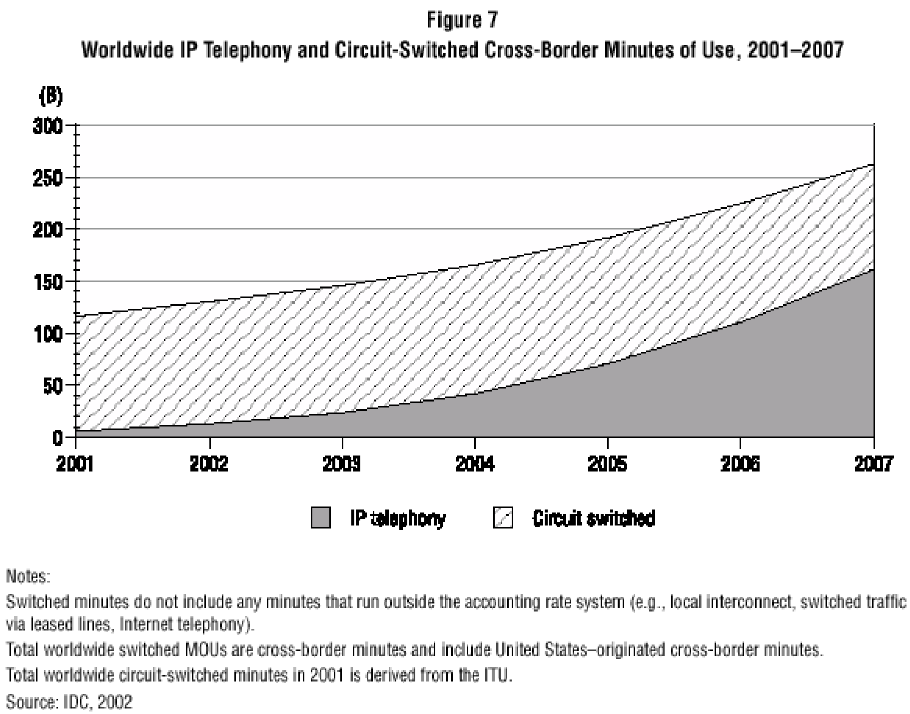
Figure 8. International VoIP Minutes of Use



This puts the IP diffusion rate for international traffic at ~20% in 2003, which is similar to the data in Reference 1. Domestic long distance and local traffic have lower rates of IP penetration, but it is also growing in those areas.

Data found in Reference 5 also roughly corresponds with previous findings, as shown in Figure 9.

Figure 9. Worldwide IP Telephony and Circuit-Switched Cross-Border Minutes of Use, 2001-2007



Note that raw VoIP MOU also may not reflect the full potential for VoIP for Service Providers, because they also use gateways to terminate PSTN lines and use IP for back-end media processing, in order to take advantage of the distributed media processing architectures and other VoIP features. This further increases the market potential for IP media servers and gateways for Service Providers.

1.3 Cost Reduction

With cost reduction being the primary factor in making business decisions in the industry, VoIP plays an important role in achieving this goal. While migrating to VoIP requires an initial investment, long-term cost savings from using it are substantial and come from a few sources:

- Maintenance
- Support
- Ease of scaling up in port density and adding new functions/applications
- Distributed application architecture is natural for VoIP and provides major advantages in deploying solutions in a flexible way

According to Reference 8, using IP PBXs in Enterprise leads to reducing costs by 40% for the systems with below 200 users and 11% for the systems with over 200 users, including the purchase and first year of maintenance.

Similar savings apply to Service Providers, although in this case the advantages are limited because of the vast existing infrastructure that is harder to change then for the Enterprise. Still, cost savings are the primary factor behind migration to VoIP as noted in Reference 9.

1.3.1 VoIP Applications

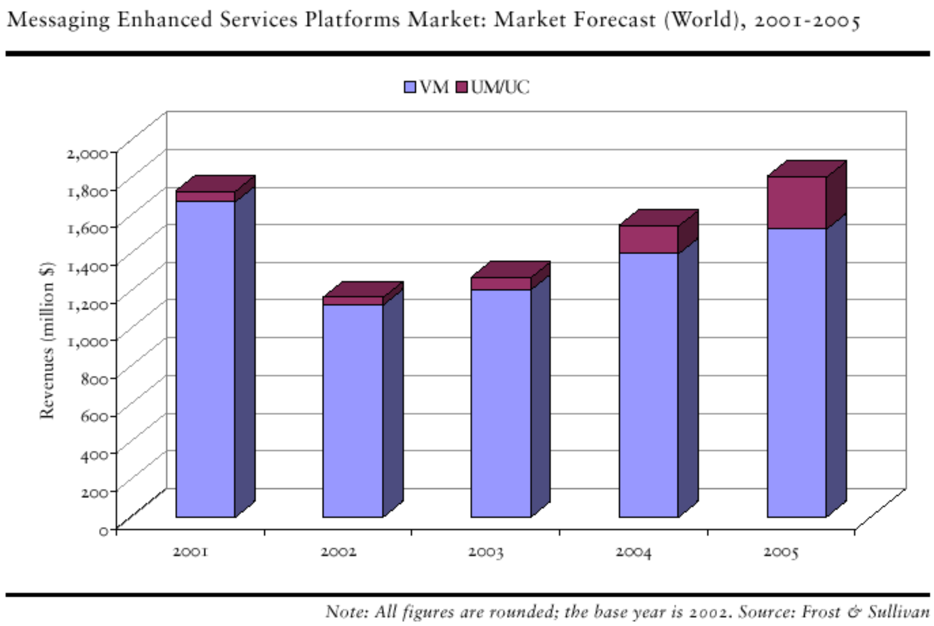
Using IP PBXs in Enterprise environments requires deploying IP media servers for such services as IVR, auto attendant, voice mail, unified messaging/unified communications.

Reference 3 states that while enhanced services (excluding voice mail) represent 13% of total enterprise services revenues for Service Providers in 2001, they are expected to grow to 20% by 2007.

For Service Providers, VoIP is primarily used today for toll bypass, especially international - 50% of all VoIP traffic is prepaid, according to Reference 10. However, as VoIP infrastructure evolves, Media server applications are growing in popularity in the Service Provider segment, especially for enhanced services platforms that include voice mail/unified messaging/unified communications, voice portals and conferencing servers. They are expected to grow rapidly within the next few years. For example, conferencing is growing at the rate of 40% and Voice Portals are expected to represent about 50% of all enhanced services revenue for Service Providers by 2006 (Reference10). Other categories such as network IVR/announcements servers and network-based IP PBXs are also growing, Reference12.

Today, voice mail still represents the majority of telecom media processing applications, because customers are primarily interested in the “nuts and bolts services”, as Reference 9 states it. The forecast for voice mail (VM) and unified communications from Reference 11 is shown in Figure 10.

Figure 10. Messaging Enhanced Services Platforms Market: Market Forcast (World), 2001-2005



However, other enhanced services such as unified messaging (UM) are expected to rapidly expand within the next two to three years, because of the following market segment drivers listed in Reference11:

- Potential increase in average revenue per user(ARPU)
- Ability to improve customer stickiness
- Untapped potential in emerging markets can fuel growth
- Higher usage of messaging in the Enterprise will lead to additional Carrier sales
- Replacement opportunity of legacy installed voice-mail base
- Competitive differentiation: gaining an edge over other Service Providers
- Pursuit of increased adoption in low penetration segments
- Consolidation of mailboxes
- Higher wireless data rates
- Future cable and interactive TV growth

1.3.2 Density requirements

As far as the solution density and scale are concerned, requirements vary depending on the application and usage model. Some considerations are:

- Cost of renting space (Service Provider)

- Cost of the processing platform
- Ability to scale the solution up within the same box (e.g. support large conferences
- Single point of failure (Service Provider)
- Ability to grow the solution (Service Provider, Enterprise)

This means that for both Enterprise and Service Provider customers, both large-scale and small-scale solutions are required. For some of the applications discussed above, the breakdown is shown in the following tables.

Reference 9 states that the most popular IP PBX size penetration has been 50 to 200 users.

Table 1 (from Reference 2) shows the density breakdown for the Enterprise enviroment.

Table 1. Enterprise Density Breakdown

2-40 ports	41-100 ports	101-400 ports	401+ ports	Total
152,439	206,744	108,232	46,200	513,615
30%	40%	21%	9%	100%

This data shows that the majority of Enterprise ports (70%) is shipped with densities of less than 100 ports per system.

Table 2 (from Reference 12) shows the number of IVR ports shipped to the United States.

Table 2. Number of IVR Ports Shipped to the United States

	Small IVR Systems (1 to 8 Ports)		Midsize IVR Systems (9 to 64 Ports)		Large IVR Systems (More Than 64 Ports		Total IVR Systems
	Number	%	Number	%	Number	%	
New ports in 2000	10,267	51%	8,688	43%	1,285	6%	20,240
Total ports in 2000	48,189	12%	201,064	50%	155,563	38%	404,816

This data is shows that 62% of all IVR ports ship with densities below 64 ports per system, which is consistent with the data from previous sources. More importantly, new port shipments represent a shift towards lower densities.

1.4 Summary of Business and Market Segment

Based on the data shown above, we can conclude that VoIP is about to rapidly accelerate in the Enterprise segment, with major new deployments in the 2004 timeframe.

Regarding the Service Provider environment where VoIP is primarily used for toll bypass today and the telecom downturn is not over yet, adoption of VoIP may be somewhat slower than in the Enterprise segment. However there still are major market segment drivers behind VoIP for Service Providers, and there is very little doubt that it will also accelerate significantly.

As for port density, the majority of Enterprise media servers require densities below 100 ports per system. Service Providers have also started considering lower port densities because of the new architectures available with the distributed deployment model.

While voice mail and audio conferencing represent the majority of media server opportunities today, new enhanced services will evolve as VoIP penetration increases.

Software-only approach to creating IP media servers is being recognized in the industry for its numerous advantages primarily in the area of cost savings. In this respect, Intel® NetStructure™ Host Media Processing is a perfect choice for building IP media servers for IVR and conferencing for densities below 100 ports per system, which represent a significant potential, primarily in the Enterprise market segment.

1.5 References

1. Advisory, Insight and Market Strategy (AIMS) "Service Report -Voice Over Packet Markets", Probe Research, Vol 3, Number 6, 2002
2. World Enterprise IP Telephony Markets, Frost & Sullivan, A167-62, 2002
3. Advisory, Insight and Market Strategy (AIMS), "Reaching the Big Guys + Global Enterprise Forecast", Probe Research, Vol 3, Number 7, 2002
4. The Imminent Demise of Global Circuit-Switched Voice Services: Greatly Exaggerated, Lisa Pierce and Brownlee Thomas, 4/26 2002, Giga Information Group, Inc.
5. IP Telephony Market Forecast and Analysis, 2002-2007, Analyst: Elizabeth Farrand, IDC, 2002
6. Telecom Spending Outlook: Looking For Opportunity In 2003, IDC Executive Telebriefing November 7, 2002, IDC Telecom Black Book Team
7. Networks and Telecom Strategy Overview, VoIP Services Snapshot, v1, The Burton Group, 04 July 2002
8. IP PBXs and IP Centrex: Market Update, Sales, Drivers, and Inhibitors; Joe Gagan, The Yankee Group, E-Networks & Broadband Access REPORT Vol. 2, No. 18-December 2001
9. 2002 World VoIP Equipment Market, Frost & Sullivan, 6747-61, 2002
10. The Media Server: The Voice of Enhanced Services- The CALEA Debate Continues. Probe Research, v. 3, #3, 2002
11. World Messaging Enhanced Services Platform Markets, Frost & Sullivan, 6792-62, 2002
12. Market Analysis, U.S. IVR Systems Market Overview, D. Kraus/L. Yu, Gartner Dataquest, 2001

Host Media Processing Configuration Description 2

Intel® NetStructure™ Host Media Processing (HMP) software makes it possible for you to build computer telephony voice applications based on the open standard IP and HMP's media processing capability.

To illustrate how to build voice applications that take advantage of HMP software, we provided a reference design of an interactive voice response (IVR) system. This IP-based IVR system has all the traditional IVR functions, but also uses IP as the underpinning technology.

This chapter includes the following information:

2.1 Solution Configuration Description

Interactive voice response (IVR) is one of the most popular Computer Telephony applications. IVR allows users to receive information via a telephone. In a typical IVR system, the user will call a service provider's number and be greeted by a pre-recorded voice prompt. The user can then respond to the IVR system by sending DTMF tones to the IVR server. The IVR server will respond interactively to these DTMF tones, based on its predetermined logic.

IVR is used in many applications such as:

- Banking
- Brokerage
- Airport flight information
- Shipping status checking
- Shopping and booking
- Directory assistance

2.1.1 Key Features of an IVR System

A typical IVR system usually has the following features:

- **Call termination if the caller's request will be handled without transferring the call to another network** - The caller will call the number and the call will be directed through a PBX or automatic call distributor (ACD) to the IVR server. Through the interaction between the caller and the IVR system, mainly done by DTMF tone generation and detection, the system will provide information based on the request from the user. For example, a user can enter the DTMF number "1" to get his or her banking account balance. If the user is satisfied with this information, he or she will then hang up the phone. In this case, the call is terminated at this server.

- **Call completion if the caller call will be transferred to a second network for more services** - In this case, the caller can't get the information requested from the local server and would like to talk to a live agent. The IVR system will forward the call to an agent through an out-bound call. The call will be terminated with a separate agent and not with this IVR system.
- **Playback of pre-recorded or dynamically constructed voice prompts and messages** - The IVR system provides information to the caller by play the information as a recorded voice file. It is possible that the IVR system dynamically concatenates a message by using voice segments. For example, the IVR system may retrieve files from a disk: "one thousand", "one hundred", "and", "one" and "dollar" to form the message "one thousand, one hundred and one dollar".
- **Record user's messages** - It is also possible for the system to record a voice message by the user. It will store the message in various voice formats such as pulse code modulation (PCM). The recorded file may be stored in a data base for further processing.
- **Interact with a database that will provide users with requested information** - Usually the database application runs on a separate application server and the two systems will be connected over a LAN.
- **Interact with other telephony servers such as automatic call distributors (ACDs) and administration servers** - This is how the IVR system receives an in-bound call and transmits out-bound call.
- **Collect information from incoming calls such as automatic number identification (ANI)** - As part of an intelligent network, the IVR system may be able to collect the ANI to identify the caller and process the call information even before the call was connected.

2.1.2 Speech Enabled IVR System

A speech enabled IVR system offers speech recognition, either based on discrete digits or natural language. The latest Intel® continuous speech processing (CSP) building blocks have been used in such applications. This reference design guide will discuss the conventional tone-based IVR systems.

2.1.3 IP-Based IVR System

An IP-based IVR system provides the above-mentioned function. However, the intrinsic nature of IP also requires special treatment when such systems are built. For example, dual tone multi-frequency (DTMF) digits can be sent both in band or out-of-band.

2.1.4 HMP 1.0 Features

HMP 1.0 provides the following features.

- Supported Codec for IP Encoding/Decoding: G.711 (64 Kb/s format) – µ law and A law

- H.323 IP Call Control Support via Global Call, as well as the ability to interoperate with third party host-based IP call control protocols using the media control APIs
- IP Media Protocol: RTP, frame sizes of 10, 20, or 30 msec
- RFC2833 Support with IP Media Library
- R4 APIs:
  - DCB Conferencing API (dcb_)
  - Dx Media API (dx_, etc.)
  - R4 Media Processing APIs for voice devices
  - IP Media Library (IPML) API to support integrating with third-party protocol stacks for call control over IP (ipm_)
  - Global Call API for network call control (H.323 IP call control only) (gc_)
  - SRL API for event handling (sr_)
- Tone Management:
  - In-Band DTMF detection/generation
  - Out-of-Band DTMF detection/generation:
    - RFC2833 via IPML
    - H.245 User Input Indication (UII) via Global Call extension
  - User-defined Global Tone Detection (GTD) and Global Tone Generation (GTG)
- Player/Recorder Formats:
  - G.711 μ Law and A Law (48 K and 64K)
  - OKI ADPCM (24K and 32K)
- Play/Record Capabilities:
  - Playing from files or buffers over IP
  - Recording from IP into files or buffers
  - Playing and recording files in all supported encoding formats with Wave headers
  - Automatic Gain Control
  - Volume Control
- Audio Conferencing
  - Up to 64 parties per server max in any number of conferences
  - Flexible conferencing
  - Hidden training
  - Individual volume control
  - Active Talker status
  - On-board digit detection with tone clamping
  - Monitoring
  - Coach/Pupil monitoring
  - Full duplex
  - High port density
- Demos:
  - IVR Demo: IPTMail_R4 Demo (voice mail functionality only)
- SNMP support – The following MIBs are supported:
  - Virtual Board (Hardware) MIBs

– R4 MIB
– DM3 Extended MIB

In addition, the current HMP release supports the following IP packet frame sizes:

- 10 milliseconds (ms)
- 20 ms
- 30 ms

In general, smaller frame sizes lead to lower latency. However, a smaller frame size will increase network traffic and CPU utilization rate. The user can experiment with each option and choose the one with the best end user experience.

For more information about HMP software architecture and capability, refer to the *Next Generation Media Processing for the Modular Network* white paper, which can be found at:

http://www.intel.com/network/csp/resources/white_papers/7786web.htm

2.2 Application Scenarios

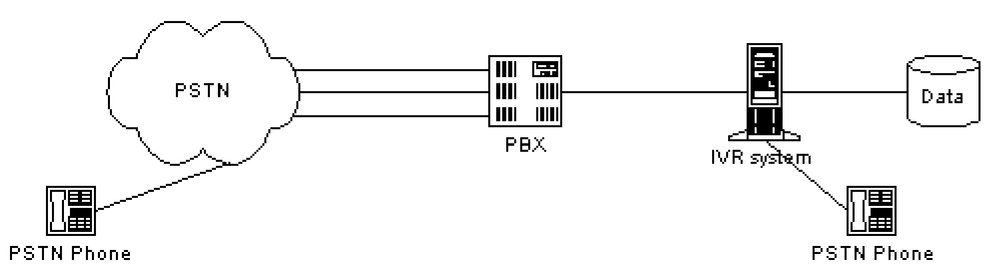
This section discusses IVR application scenarios:

- PSTN Network
- HMP Based IVR Systems

2.2.1 PSTN Network

The following diagram shows the traditional IVR system which is based on the PSTN network.

Figure 11. PSTN Based IVR System



In this system, the user calls the number and is connected to the IVR system. The IVR system collects the DTMF digits and then retrieves information from the database. The IVR system usually has the capability to transfer the caller's call to an agent if it is needed.

2.2.2 HMP Based IVR Systems

The HMP-based IVR system has three options:

1. Option One: The caller will use an IP phone to interact with the IVR system, Figure 12.
2. Option Two: The caller can still use a PSTN phone, but the phone will be connected through an access gateway to the IVR system, Figure 13.
3. Option Three: The caller accesses the IVR system via a wireless network, where again a gateway resides between the HMP-based IVR system and the wireless network.

Note: A gateway is required somewhere in the network in order to have a connection between the HMP-based IVR system and a PSTN-based user.

Figure 12. HMP Based IVR System - Option 1

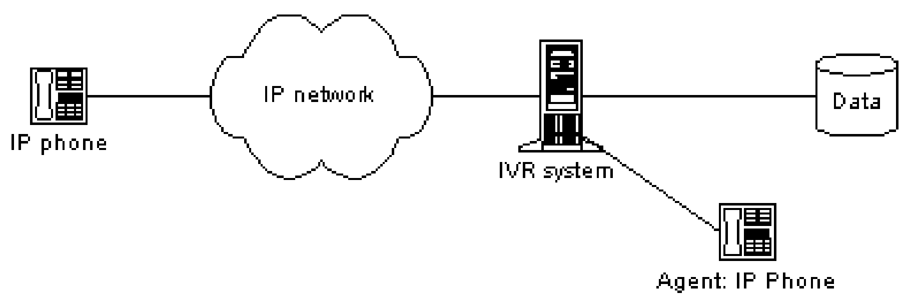
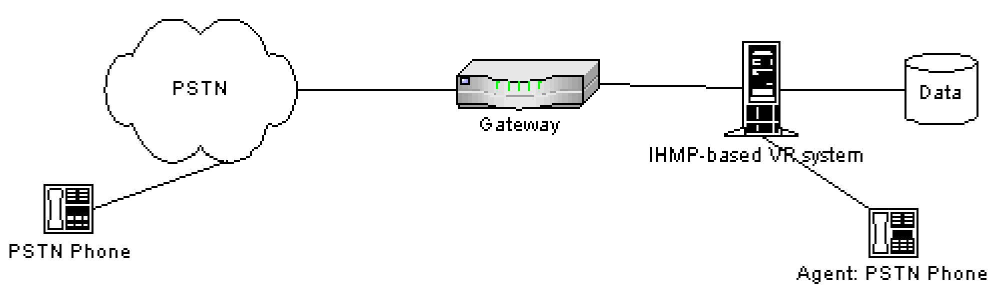


Figure 13. HMP Based IVR System - Option 2



2.3 Equipment Tested

HMP has been tested with the following equipment:

- IP Phones:
  - Polycom SoundPoint IP 400

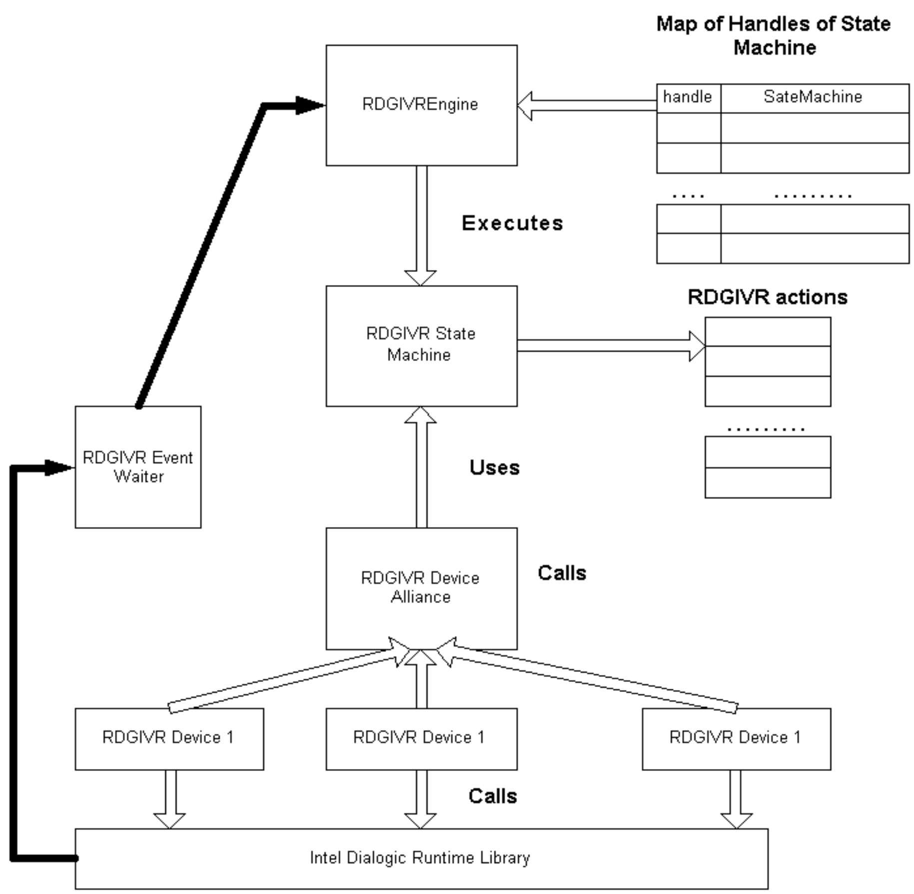
  - Siemens optiPoint 400 standard v.3.0
  - Cisco 7960
- Gateways:
  - Cisco 5300
  - Intel® NetStructure™ PBX-IP Media Gateway

2.4 Architecture of the Software Reference System

Figure 14, “Software Reference System Architecture”, on page 41 depicts the architecture of the reference code that is available online at this URL:

http://membersresource.intel.com/_mem_bin/FormsLogin.asp?/search/ddl/download/DDLResults.asp?Genre=Download%20--%20SDK&PKey=HMP%20Software&SKey=IP%20Media&OrderBy=Ascending&SortBy=Title

Figure 14. Software Reference System Architecture



Installing the Host Media Processing Software — 3

This chapter contains relevant excerpts from the *Intel® NetStructure™ Host Media Processing Software Release 1.0 for Windows Installation and Configuration Guide*, to which you can refer for optional procedures and more details. The information in this chapter is organized as follows:

3.1 Installing HMP Software

Perform the following steps to install the HMP software. The software will be installed in */usr/dialogic*.

1. Exit all other programs you may have running.
2. Insert the HMP CD-ROM in your system.

   If the installation process does not start automatically when you insert the CD-ROM, locate the *Setup.exe* program on the CD-ROM and double-click on the filename.

   Note: If you are running Terminal Services on Windows 2000, you must put a terminal server in install mode before you install a program. To do this, you must use **Add/Remove Programs** in the **Control Panel**.

3. The Welcome screen appears. Click **Next**.
4. The License Agreement appears. Read the agreement. You cannot continue with the installation unless you click **Yes** to accept the agreement. If you click **No**, you will exit the installation.
5. The Customer Information screen appears. Enter your name and company and click **Next**.
6. The Choose Destination Location screen appears. To install to the folder already named on the screen, click **Next**. To install to a different folder, click **Browse** and select another folder. Then click **Next**.
7. The Select Components screen appears. Select IP Media Server and click **Next**.

   An **IP Media Server** connects to the IP network over the Ethernet and operates on media streams that have already been processed (packetized, echo cancelled, etc.), by a gateway device. The server will interface to the network exclusively over a high-speed packet interface.

8. On the next screen, you choose how you'd like to access the documentation that supports HMP. You can install the documentation on your computer or elect to access it from the CD whenever you need it. Click **Next**.

   Note: The documentation is in PDF format. To access it, Adobe* Acrobat Reader* (v3.01 or later) must be installed. You can get Adobe Acrobat Reader from the Adobe web site: *http://www.adobe.com/products/acrobat/readstep2.html.*

9. The Enter IP Address screen requests your machine's IP address. If you don't know the IP address, open a DOS command prompt window, type `ipconfig`, and press **Enter**. The IP address will then be displayed. Type this IP address in the IP Address screen and click **Next**.

   Note: HMP uses a static IP address that is stored in the registry. If the IP address of your system changes after you install the license, you need to edit the IP address in the registry (HKEY_LOCAL_MACHINE\SOFTWARE\SBLabs\dm3ssp\IP_Addr0) and then restart the Intel® Dialogic® System Service.

10. The Select Program Folder screen shows the Program Folder where the installation will add program icons. You can type a new folder name or select one from the existing folders list. Click **Next** to continue.

11. The Start Copying Files screen shows you a summary of the settings you've selected. If you're satisfied with them, click **Next** to start copying files. If you want to make changes, click **Back** to go to a previous screen.

12. A status bar will show the progress of the installation.

13. The InstallShield Wizard Complete screen gives you the option to reboot your computer now or later. You must reboot before you can use HMP. Make your selection and click **Finish**.

After your computer reboots, an Information screen will tell you what to do next. You can do either of the following:

- **Run the demo program**. Refer to the *IPTMail_R4 Demo Guide*.
- **Purchase and activate a permanent HMP license**. To do this, start the HMP License Manager, which is available via **Start > Programs > Intel NetStructure HMP 1.0 > HMP License Manager**. Refer to Chapter 4, "HMP Licensing Procedure".

After you have purchased and activated a permanent HMP license, start the Intel® Dialogic® Configuration Manager (DCM) via **Start > Programs > Intel NetStructure HMP 1.0 > Intel Dialogic Configuration Manager - DCM**. DCM will detect HMP and automatically find the licensed configuration settings. DCM also allows you to start the Intel Dialogic System. For more details, refer to Chapter 5, "Configuring the Host Media Processing Software".

Note: You can put the HMP License Manager and DCM icons on your desktop so that you don't have to access them from **Start > Programs**.

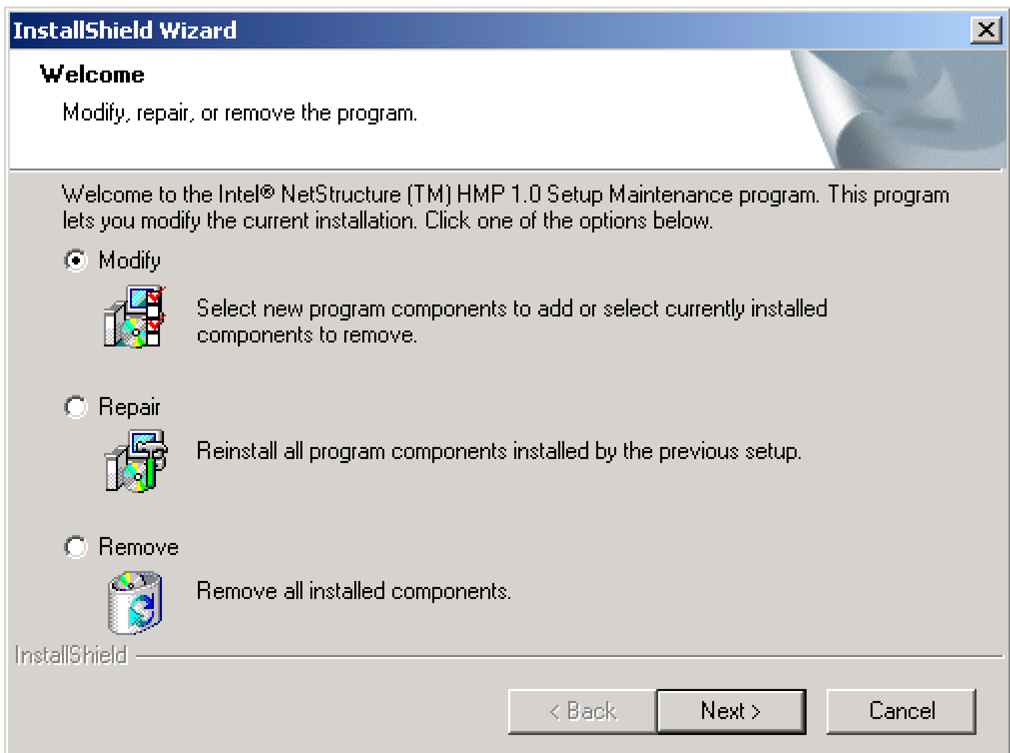
Now you can proceed to the configuration process by referring to Chapter 5, "Configuring the Host Media Processing Software".

3.2 Uninstalling HMP Software

If you should need to modify an installation by adding or removing components, reinstall HMP, or uninstall HMP, follow these steps:

1. From the Start menu, select **Start > Programs > Intel NetStructure HMP 1.0 > Uninstall**. An InstallShield Wizard screen appears (Figure 15).

Figure 15. Modify, Repair, or Remove



2. Select the option you want (**Modify**, **Repair**, or **Remove**) based on the description on the screen.

3. Click **Next**. The InstallShield Wizard will do the action you requested.

HMP Licensing Procedure 4

This chapter provides detailed information about HMP licensing and includes the following procedures:

Note: For information about HMP features, applications, licensing, the free demo license, configurations, functional description, technical specifications, hardware system requirements, and operating system requirements, refer to the Intel® NetStructure™ Host Media Processing software information at *http://www.intel.com/network/csp/products/8160web.htm.*

4.1 Obtaining the Evaluation Run-Time License

A four-port trial license is available for evaluation purposes that will allow you to enable an HMP configuration for a limited duration. This configuration consists of four voice processing channels. This license automatically terminates at the end of the evaluation period.

To obtain this trial license, perform the following:

1. Go to the following Intel Web site:

   http://www.intel.com/network/csp/products/hmp/8399web.htm#licensing

2. After reading the terms of the license agreement, click the **I accept** button.

3. A customer registration Web page will appear. Enter the required information and click the **SUBMIT** button.

   The four-port trial license will be emailed to you.

4.2 Purchasing a Permanent HMP License

This procedure describes how to use the HMP License Manager to purchase an HMP license.

1. Access the HMP License Manager from the Start menu as follows: **Start > Programs > Intel HMP 1.0 > HMP License Manager**. The HMP License Manager appears (Figure 16, “HMP License Manager”, on page 48).

2. On the HMP License Manager window (Figure 16), click the **How to Purchase License...** button. The HMP Purchase License Wizard will appear (Figure 17). This wizard will guide you through the steps necessary to purchase a permanent HMP license.

   Note: The wizard does not automatically generate a license file. Instead, it provides instructions on how to purchase a license from your vendor and how to activate the license.

Figure 16. HMP License Manager

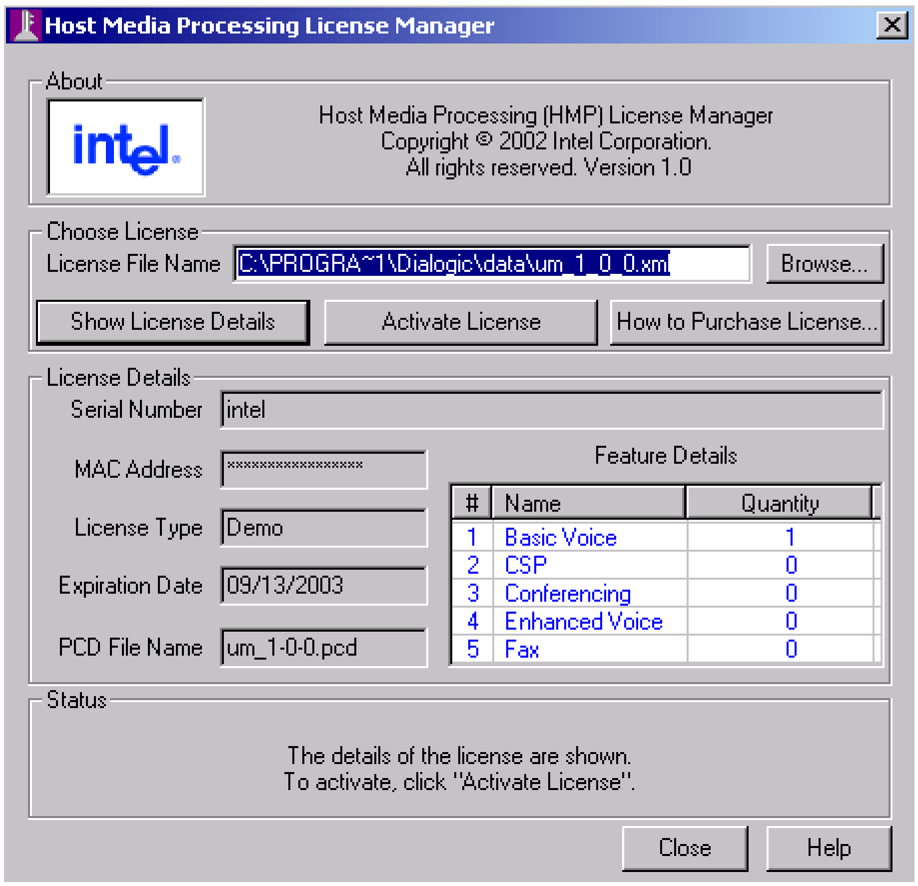
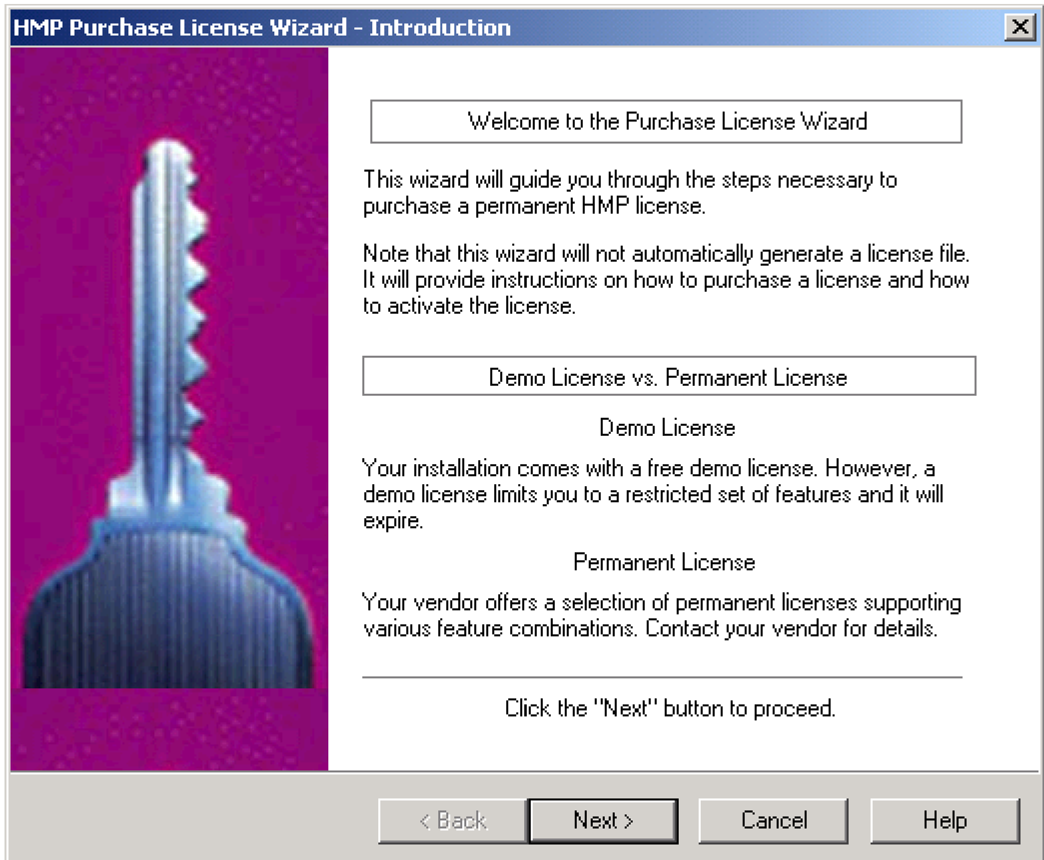
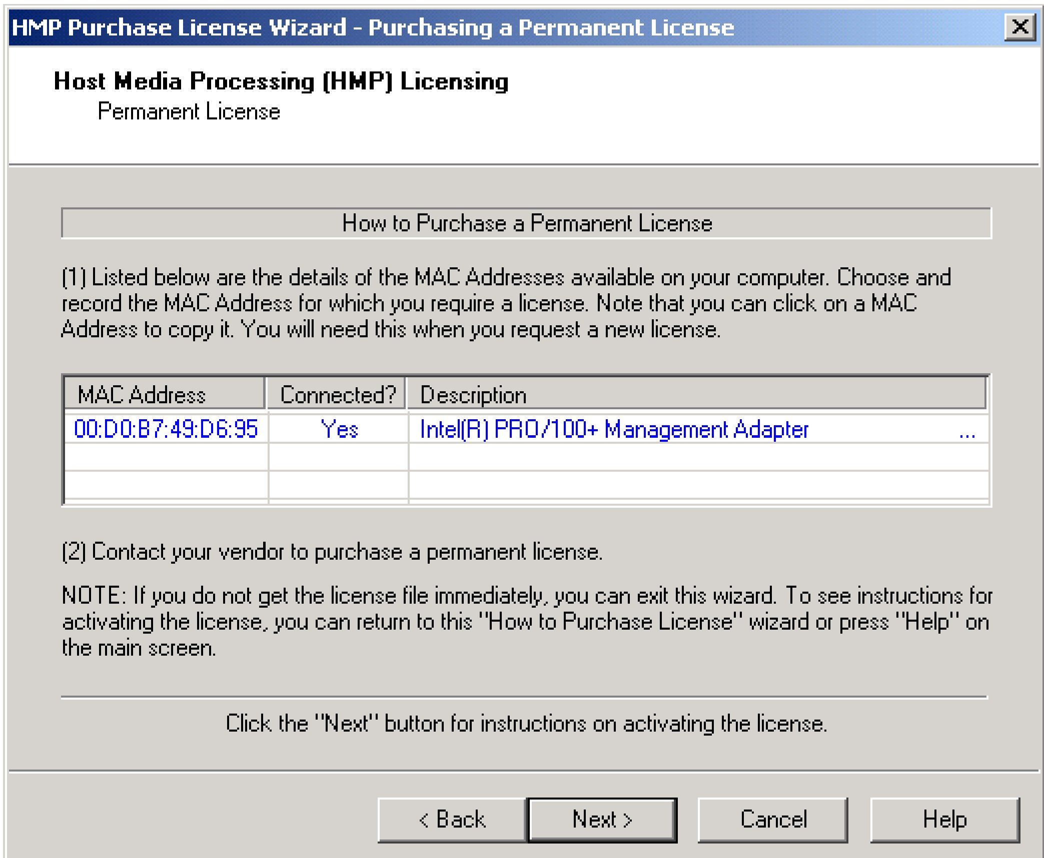


Figure 17. Purchase License Wizard - Introduction



3. After reading this introductory screen, click **Next**. The second screen of the wizard appears (Figure 18).

Figure 18. Purchase License Wizard - Purchasing a Permanent License



4. As instructed on the screen, choose and record the MAC address for which you require a license.

5. Contact your vendor to purchase a permanent license. You will need to provide the MAC address you obtained in Step 4. For a list of available licenses, refer to Table 3, “HMP 1.0 SKUs and Solution Configurations”, on page 51.

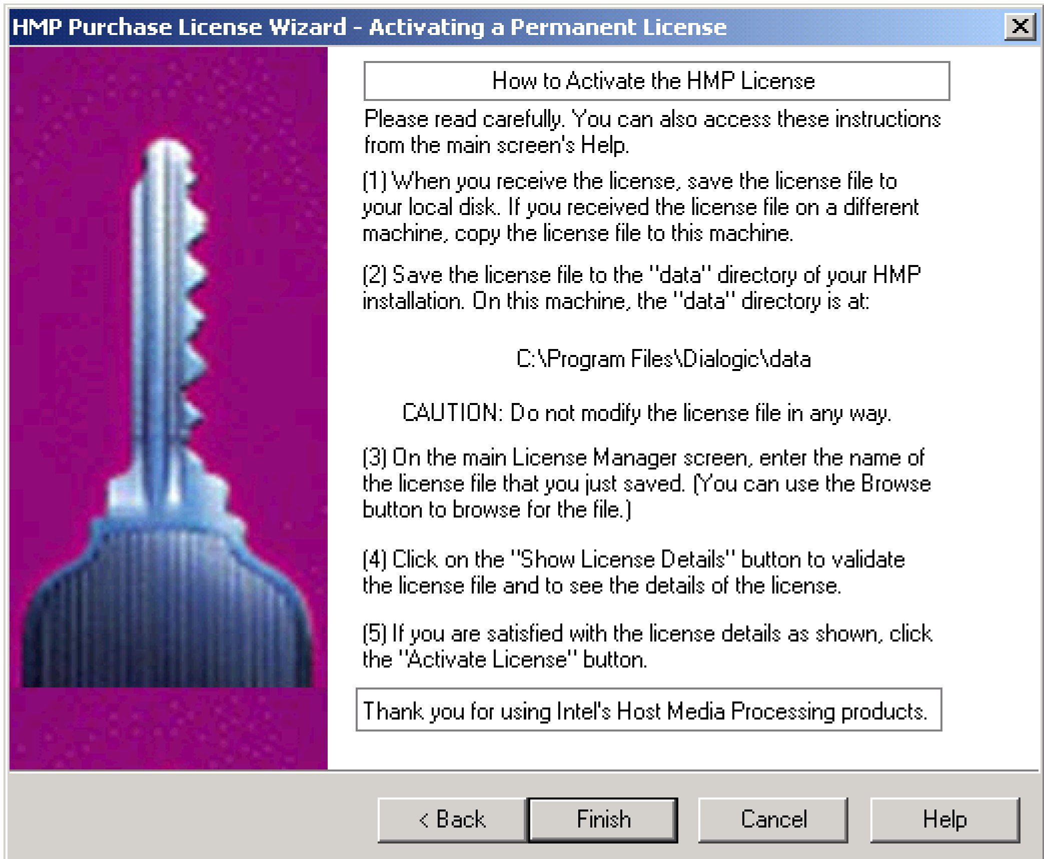
Table 3. HMP 1.0 SKUs and Solution Configurations

Data Sheet Product Name	HMP Configuration	Voice Processing Channels	Conferencing Channels	RTP Sessions
DMIPS40AW	umco_4-4-0	4	4	4
DMIPS80AW	um_8-0-0	8	0	8
DMIPS160AW	um_16-0-0	16	0	16
DMIPS320AW	um_32-0-0	32	0	32
DMIPS640AW	um_64-0-0	64	0	64
DMIPS321AW	umco_32-32-0	32	32	32
DMIPS641AW	umco_64-64-0	64	64	64
DMIPS322AW	umco_32-16-0	32	16	32
DMIPS480AW	umco_48-16-0	48	16	48

Note: When you purchase a license, all licenses with fewer features than the license you have bought are also available for your use. For example, if you purchase a umco_48-16-0 license, you can also use all the licenses listed above umco_48-16-0 (with fewer features) in Table 3. However, the Intel® Dialogic® Configuration Manager (DCM) will automatically detect the license with the highest number of features. So if you want to use one of the others, follow the procedure in Section 4.4, “Using a Different License”, on page 53.

If you do not get the license file immediately, you can exit the wizard (click **Cancel**) or look at the instructions for activating the license by clicking **Next**. When you click Next, the third screen of the wizard appears (Figure 19).

Figure 19. Purchase License Wizard - Activating a Permanent License



4.3 Activating an HMP License

When you receive the license, follow the activation instructions, which are provided in the online help for the HMP License Manager GUI and on the third screen of the HMP Purchase License Wizard (Figure 19). The activation procedure is also given here:

1. When you receive the license from your vendor, save the license file onto the computer containing HMP, in the %DLFWLPATH% directory. This is the "data" directory of your HMP installation. The third screen of the HMP Purchase License Wizard will show you the path to this directory on your machine (for an example, see Figure 19). Another way to find out your %DLFWLPATH% directory is to type `echo %DLFWLPATH%` on a command prompt and note down the path displayed.

CAUTION

Do *not* modify the license file in any way.

2. If the HMP License Manager GUI is not open on your screen, access it via **Start > Programs > Intel HMP 1.0 > HMP License Manager**.

3. On the HMP License Manager screen (Figure 16), enter the name of the license file that you just saved. (You can use the Browse button to browse for the file.)

4. Click on the **Show License Details** button to validate the license file and to see the details of the license.

5. If you are satisfied with the license details as shown, click the **Activate License** button.

Now you can start the Intel Dialogic Configuration Manager (DCM), which will detect HMP and automatically find the configuration settings.

4.4 Using a Different License

Your computer can only use one license at a time. If you want to switch to using a different license, follow the procedure in this section.

Note: When you purchase a license, all licenses with fewer features than the license you have bought are also available for your use. For example, if you purchase a umco_48-16-0 license, you can also use all the licenses listed above umco_48-16-0 (with fewer features) in Table 3. This procedure tells you how to activate a different license.

1. If you're buying a new license (with more features) to replace an old license (with fewer features), start by following the procedure in Section 4.2, "Purchasing a Permanent HMP License", on page 47. Otherwise, skip to step 2.

2. If the HMP License Manager GUI is not open on your screen, access it via **Start > Programs > Intel HMP 1.0 > HMP License Manager**.

3. On the HMP License Manager screen (Figure 16), enter the name of the license file that you want to use. (You can use the Browse button to browse for the file.)

4. Click on the **Show License Details** button to validate the license file and to see the details of the license.

5. If you are satisfied with the license details as shown, click the **Activate License** button.

6. Start DCM as described in Chapter 5, "Configuring the Host Media Processing Software".

7. On the DCM Main Window (Figure 21) click the **Stop Service** option from the **Service** pull-down menu or click the **Stop Service** icon.

8. From the DCM Action pull-down menu, click **Restore Defaults**. DCM will detect HMP and automatically use the new license.

Configuring the Host Media Processing Software 5

This chapter contains the following information about configuring the Host Media Processing Software (HMP) and running the solution:

5.1 Configuration Overview

HMP is configured using the Intel® Dialogic® Configuration Manager (DCM). This chapter describes the DCM and how to start it. DCM provides a graphical user interface for viewing and modifying configuration data. When you start DCM, it detects HMP and automatically finds the configuration settings.

5.2 Configuration Procedure

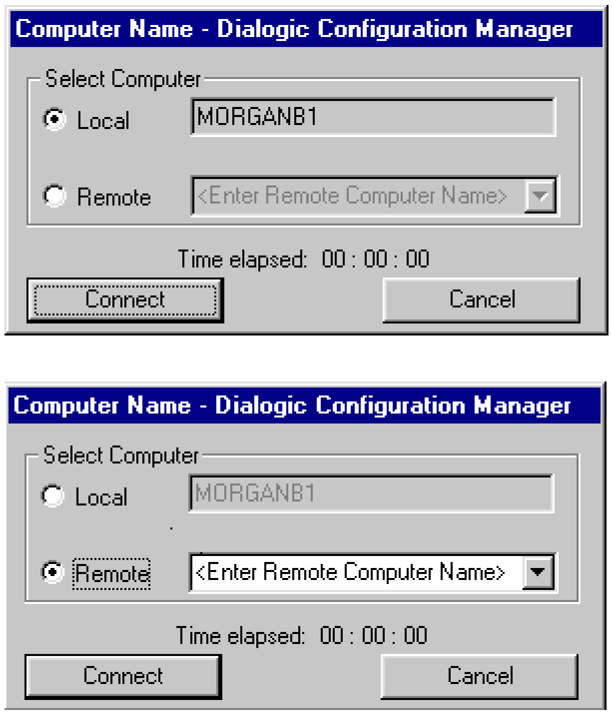
To start DCM, follow these steps:

1. Access the Intel Dialogic Configuration Manager (DCM). From the **Start** menu, select **Programs** > **Intel HMP 1.0** > **Configuration Manager-DCM.**

   Note: To use DCM, you must have administrative privileges on your local computer and on any remote computer you connect to. Contact your network administrator to set up administrative privileges as required.

2. When DCM starts, specify the computer to connect to in the **Computer Name** dialog box (Figure 20). It will display automatically the first time you run DCM. If the **Computer Name** dialog box isn't already displayed, you can get it by selecting the **Connect** option from the **File** pull-down menu or by clicking the **Connect** icon on the DCM Main Window.

Figure 20. Computer Name Dialog Box



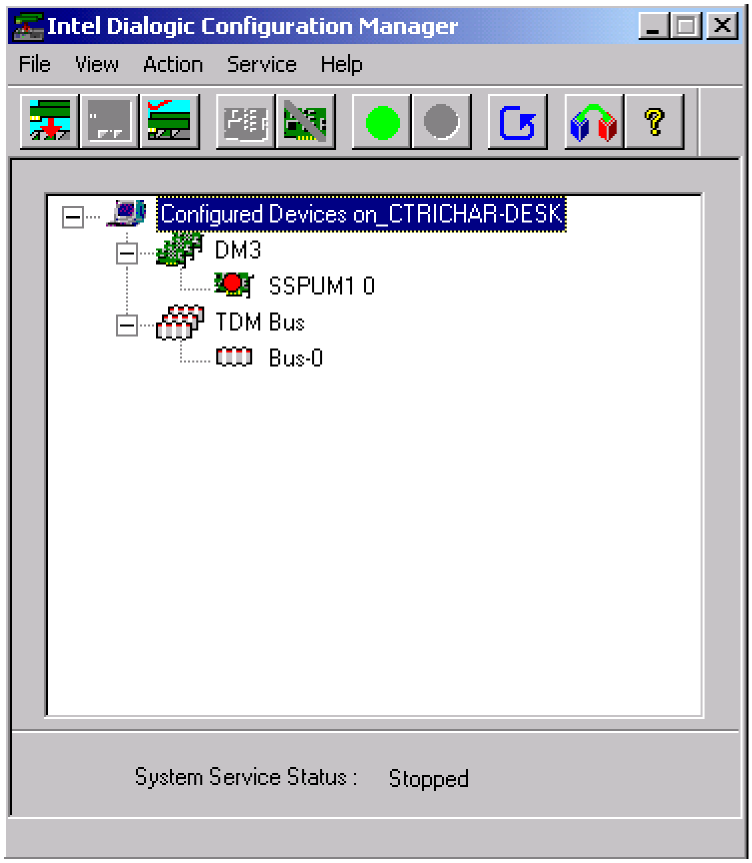
3. The local computer name is displayed by default. To connect to the local computer, click **Connect**. To connect to a remote computer, select the **Remote** radio button, enter the remote computer name, and click **Connect**. For TCP/IP networks, you can enter the IP address instead of the remote computer name.

   Note: The Intel Dialogic System uses DCOM objects to run Intel Dialogic software on remote computers. Remote DCM Intel Dialogic software internally sets up the DCOM security level programmatically. Do *not* use the Windows DCOM configuration utility *dcomcnfg.exe* to change the security settings. If you do, the Intel Dialogic System may not work properly.

Note: After you connect to a computer, a window will appear indicating that the DCM detection routine is running.When you want DCM to recognize a new license, you must run **Restore Defaults** from the Action menu so that DCM will detect the new license. Refer to Section 4.4, “Using a Different License”, on page 53.

The DCM Main Window appears (Figure 21). This display is for the computer you specified using the **Computer Name** dialog box (Figure 20). The top line of the display, Configured Devices on..., shows the name of the computer you connected to. If you entered an IP address in the **Computer Name** dialog box, the IP address is shown instead of the computer name.

Figure 21. DCM Main Window



5.3 Starting and Stopping the System

After you install HMP, purchase a permanent HMP license using the License Manager GUI, and start DCM, you can start the Intel Dialogic System. This downloads configuration parameter settings to HMP and initiates the device drivers. Following this, you can use some of the tools provided by Intel to verify that your system is operating properly, before starting work on your applications. This information is provided as follows:

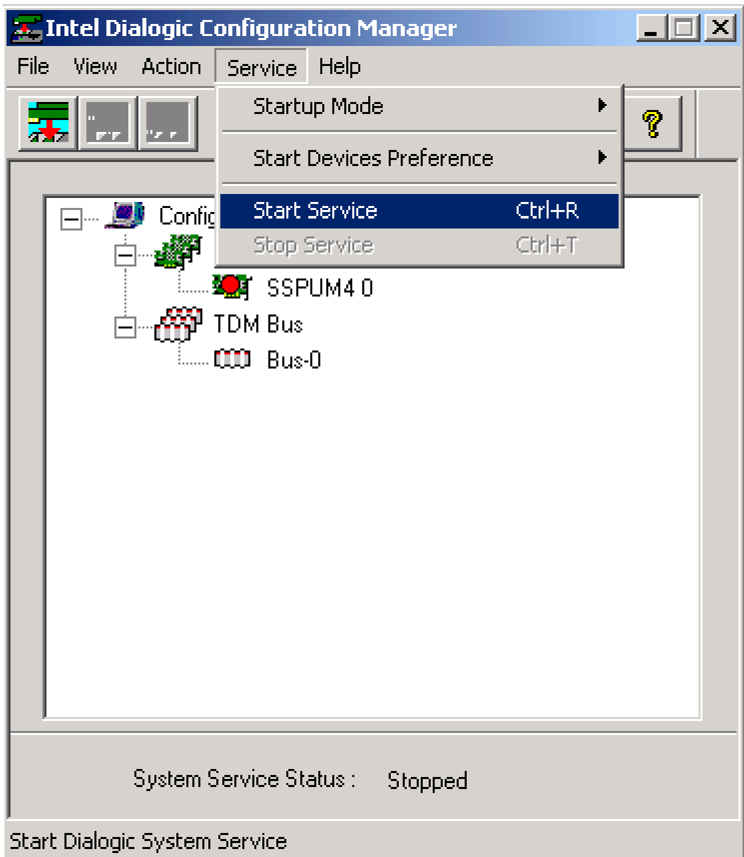
- Starting the System
- Setting Startup Mode to Automatic

5.3.1 Starting the System

To start the system, click the **Start Service** option from the DCM **Service** pull-down menu or click the **Start Service** icon on the DCM Main Window (Figure 21).

Note: The Start Devices Preference option on the DCM Service pull-down menu does not apply to HMP Release 1.0 because there is only one virtual board (SSPUM4 0) in the system.

Figure 22. Start the System



DCM displays a progress bar in the upper right corner of the screen during the wait time. When the Intel Dialogic System starts, the **Intel Dialogic System Service Status** indicator at the bottom of the DCM Main Window indicates **Running**.

The progress bar is normally displayed until the Intel Dialogic System is started. However, when connected to a remote node, for example, the time to start the Intel Dialogic System could be extended because of network latency. If the progress bar is no longer displayed but the **Intel Dialogic System Service Status** indicator does not indicate the **Running** state yet, this does not necessarily indicate a problem. Click the **Refresh** icon on the DCM Main Window periodically to update the **Intel Dialogic System Service Status** indicator. Eventually, it should indicate **Running**. If not, check the Windows Event Viewer to see if an error occurred.

You cannot start the Intel Dialogic System while HMP is disabled. You also cannot start the Intel Dialogic System from the Control Panel and then open DCM.

If you need to detect devices, use DCM's **Action** menu options or the **Control Panel** to stop the Intel Dialogic System and initiate device detection.

5.3.2 Setting Startup Mode to Automatic

There are cases where the Intel Dialogic System has to be started automatically (without human assistance). Users with administrative privileges can set the **Startup Mode** to **Automatic** using the DCM GUI's **Service** pull-down menu (Figure 22) or programmatically using the NCM API library. If the **Startup Mode** is set to **Automatic**, the Intel Dialogic System restarts automatically when the system reboots.

Notes: ***1.*** Do *not* use the Windows Services applet to set the Intel Dialogic System startup mode to **Automatic**. You must use the DCM GUI or the NCM API to do this because they internally set the startup mode of the Intel Dialogic drivers to **Automatic**. If you use the Windows Services applet, you will not set up the driver dependencies properly.

2. If you set the **Startup Mode** to **Automatic**, and you want to reconfigure HMP through DCM after rebooting the computer, you should use DCM to stop the Intel Dialogic System and then do the normal operations through DCM. (It is possible to use Windows Services applet to stop the Intel Dialogic System but it is not recommended.)

Understanding the IVR Reference Application 6

This chapter provides information about the following topics:

The IVR reference application (SimpleIvr) provides demo code for Intel customers who want to build custom applications using R4 and Global Call APIs included with the software development kit (SDK). The application is developed using C++ and follows object-oriented design guidelines that allow the easy addition of new components as well as the reuse of existing components. The SimpleIvr application is designed as a single-threaded, asynchronous application to meet the necessary performance requirements, and uses some special techniques, like thread pooling, to perform lengthy operations.

The application components are driven by events produced by the SDK Standard Runtime Library (SRL) as a direct or indirect reaction to physical activities in the underlying telephony or network circuitry, as well as events posted by the application to the SRL event queue (self-posted reflection events). In this sense, one can look to the application as an example of a reactive system characterized by being event-driven and continuously having to react to external and internal stimuli.

6.1 Obtaining the IVR Reference Code

You may obtain the IVR reference code from the following Web site:

http://membersresource.intel.com/_mem_bin/FormsLogin.asp?/search/ddl/download/DDLResults.asp?Genre=Download%20--%20SDK&PKey=HMP%20Software&SKey=IP%20Media&OrderBy=Ascending&SortBy=Title

6.2 State Machines

The systems of the class described above are best implemented with the help of finite state machines. Finite state machines and their corresponding state-transition diagrams are the formal mechanism for aggregating statements such as: "when event α occurs in state A, the system transfers to state B." However, complex systems usually have a large number of distinct states with even more possible transitions between them.

That brings us to the problem of managing the state diagrams for the application. There are several ways to overcome this problem. You may develop an external state management subsystem with its own database for storage state diagrams, GUI front-ends to edit and validate them, and code generators which produce pseudo-code or even an actual programming language code that implements the state machine.

Another approach is more sophisticated and involves decomposing a large state machine into smaller relatively independent or orthogonal sub-machines (clusters) and then bridging them, thereby reducing otherwise exponential numbers of state transitions. Other methods are available to deal with the problem of large state machines, but they are outside the scope of this document and can be researched independently.

6.3 A Basic State Machine

The SimpleIvr reference application implements, and is built around, a basic state machine abstraction that is best described with AαB formalism. The state complexity is reduced by having a common state, Ω, that deals with unexpected or erroneous conditions that may occur during the runtime of the application and logs the history for future analysis and system improvements. This abstraction is represented in the application in the form of SimpleStateMachine and has a concrete implementation with SimpleIvrStateMachine class.

To illustrate how the state machine is created, a fragment of SimpleIvrStateMachine class constructor is shown in the following sample code:

```
SimpleIvrStateMachine::SimpleIvrStateMachine()
{
// Associate states with actions
addStateAction(STATE_NULL, NullAction);

addStateAction(STATE_WAIT_CALL, WaitCall);
addStateAction(STATE_CALL_ACK, CallAck);
addStateAction(STATE_ANSWER, AnswerCall);

addStateAction(STATE_INIT_PROMPT, PlayInitialPrompt);
…
addStateAction(STATE_DROP, DropCall);
addStateAction(STATE_RELEASE, ReleaseCall);
addStateAction(STATE_RESET, ResetLineDev);

// Create state transitions
add(STATE_NULL,                  GCEV_UNBLOCKED,     STATE_WAIT_CALL);
```

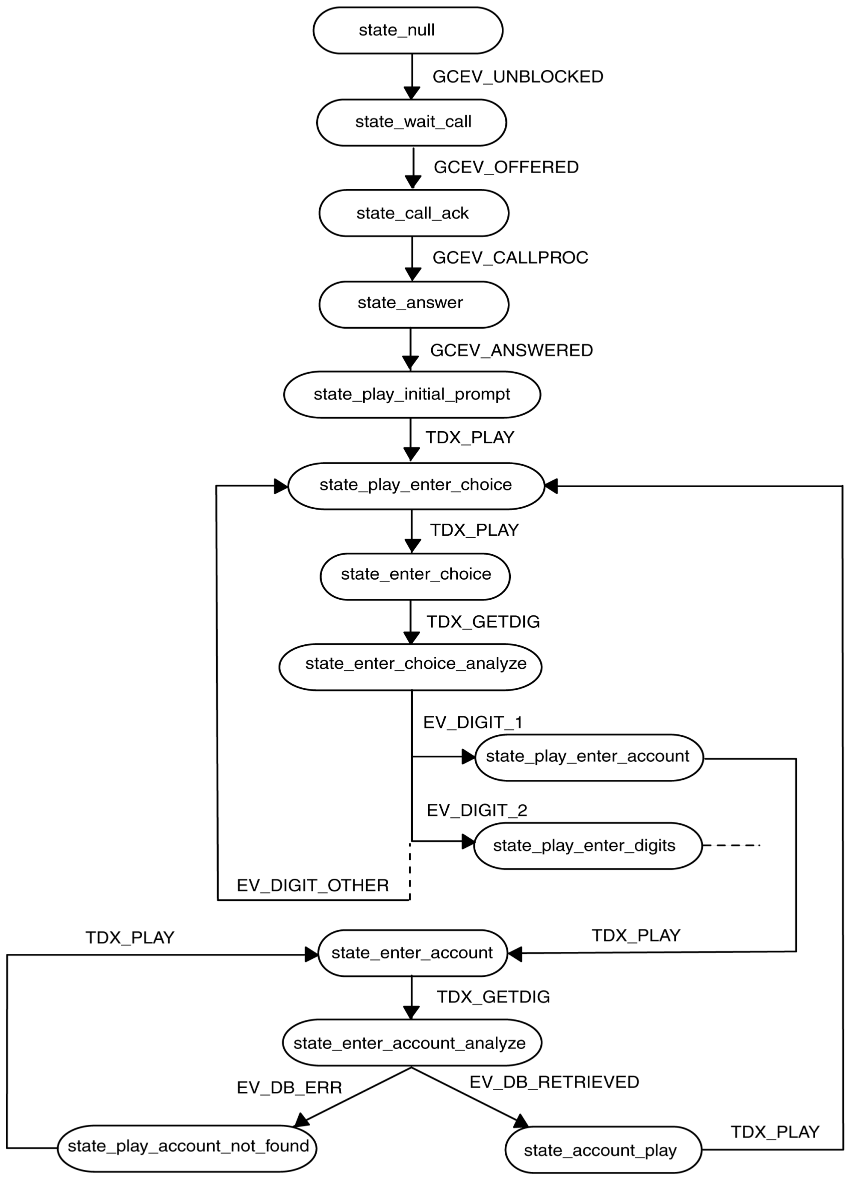
```
add(STATE_WAIT_CALL,              GCEV_OFFERED,       STATE_CALL_ACK);
add(STATE_WAIT_CALL,              GCEV_DISCONNECTED,  STATE_RESET);

add(STATE_CALL_ACK,               GCEV_CALLPROC,      STATE_ANSWER);
add(STATE_CALL_ACK,               GCEV_DISCONNECTED,  STATE_RESET);

add(STATE_ANSWER,                 GCEV_ANSWERED,      STATE_INIT_PROMPT);
add(STATE_ANSWER,                 GCEV_DISCONNECTED,  STATE_RESET);

add(STATE_PLAY_INIT_PROMPT,       TDX_PLAY,           STATE_PLAY_ENTER_CHOICE);
add(STATE_PLAY_INIT_PROMPT,       GCEV_DISCONNECTED,  STATE_DROP);

add(STATE_PLAY_ENTER_CHOICE,      TDX_PLAY,           STATE_ENTER_CHOICE);
add(STATE_PLAY_ENTER_CHOICE,      GCEV_DISCONNECTED,  STATE_DROP);

add(STATE_ENTER_CHOICE,           TDX_GETDIG,         STATE_ENTER_CHOICE_ANALYZE);
add(STATE_ENTER_CHOICE,           GCEV_DISCONNECTED,  STATE_DROP);

add(STATE_ENTER_CHOICE_ANALYZE,   EV_DIGIT_1,         STATE_PLAY_ENTER_ACCOUNT);
add(STATE_ENTER_CHOICE_ANALYZE,   EV_DIGIT_2,         STATE_PLAY_ENTER_DIGITS);
add(STATE_ENTER_CHOICE_ANALYZE,   EV_DIGIT_3,         STATE_PLAY_ENTER_MAILBOX_R);
add(STATE_ENTER_CHOICE_ANALYZE,   EV_DIGIT_4,         STATE_PLAY_ENTER_MAILBOX_P);
add(STATE_ENTER_CHOICE_ANALYZE,   EV_DIGIT_STAR,      STATE_PLAY_ENTER_CHOICE);
add(STATE_ENTER_CHOICE_ANALYZE,   EV_DIGIT_OTHER,     STATE_PLAY_ENTER_CHOICE);
add(STATE_ENTER_CHOICE_ANALYZE,   GCEV_DISCONNETED,   STATE_DROP);

add(STATE_PLAY_ENTER_ACCOUNT,     TDX_PLAY,           STATE_ENTER_ACOUNT);
add(STATE_PLAY_ENTER_ACCOUNT,     GCEV_DISCONNECTED,  STATE_DROP);

add(STATE_ENTER_ACCOUNT,          TDX_GETDIG,         STATE_ENTER_ACCOUNT_ANALYZE);
add(STATE_ENTER_ACCOUNT,          GCEV_DISCONNECTED,  STATE_DROP);

add(STATE_ENTER_ACCOUNT_ANALYZE,  EV_DB_RETRIEVED,    STATE_ACCOUNT_PLAY);
add(STATE_ENTER_ACCOUNT_ANALYZE,  EV_DB_ERR,         STATE_PLAY_ACCOUNT_NOT_FOUND);
add(STATE_ENTER_ACCOUNT_ANALYZE,  GCEV_DISCONNECTED,  STATE_DROP);

add(STATE_ACCOUNT_PLAY,           TDX_PLAY,           STATE_PLAY_ENTER_CHOICE);
add(STATE_ACCOUNT_PLAY,           GCEV_DISCONNECTED,  STATE_DROP);

add(STATE_PLAY_ACCOUNT_NOT_FOUND, TDX_PLAY,           STATE_ENTER_ACCOUNT);
add(STATE_PLAY_ACCOUNT_NOT_FOUND, GCEV_DISCONNECTED,  STATE_DROP);

...

add(STATE_DROP,             GCEV_DROPCALL,STATE_RELEASE);
add(STATE_DROP,             GCEV_DISCONNECTED,STATE_RELEASE);

add(STATE_RELEASE,          GCEV_RELEASECALL,STATE_WAIT_CALL);
add(STATE_RELEASE,          GCEV_DROPCALL,STATE_RELEASE);
}
```

The results are shown in Figure 23, "State Transition Diagram".

The driving force for all SimpleStateMachines is one and only one instance of SimpleEventWaiter object. This class implements all functionality related to SRL event management and distribution inside the application. It is also an observable object, which allows other objects in the application context to register with it and get notified about the events happening in the system. The main loop implementing this functionality is illustrated in the following code:

```
// Event listening loop until requested to stop
while (!m_bStop)
{
         // Wait for all events from SRL
         m_ret = sr_waitevt(m_tmout);
         if (SR_TMOUT == m_ret) {}

         // Platform-specific event properties retrieval
#ifdef _WIN32
         lEventType = sr_getevttype(0);
         lEventDev  = sr_getevtdev(0);
         lEventLen = sr_getevtlen(0);
         pData  = sr_getevtdatap(0);
#else
         lEventType = sr_getevttype();
         lEventDev  = sr_getevtdev();
         lEventLen = sr_getevtlen();
         pData  = sr_getevtdatap();
#endif
         waitevt = m_ret;

         // Create a SimpleEvent object based on event properties
         SimpleEvent event(waitevt, lEventType, lEventDev, lEventLen, pData);

         // Set object status to changed and notify it observers
         setChanged();
         notifyObservers(&event);
}
```

Figure 23. State Transition Diagram



6.4 Creating Objects

The main consumer of the events produced by SimpleEventWaiter is a SimpleIvrEngine object which creates and keeps all SimpleIvrStateMachine objects during the initialization phase, and distributes the events to the appropriate state machines. SimpleIvrEngine may be considered as a "main" class of the application. It is one of the first objects created, which in turn orchestrates the creation of all other supporting entities, including SimpleEventWaiter. SimpleIvrEngine then passes control to the SimpleEventWaiter and goes into a passive mode as a listener. The following code illustrates how this is implemented:

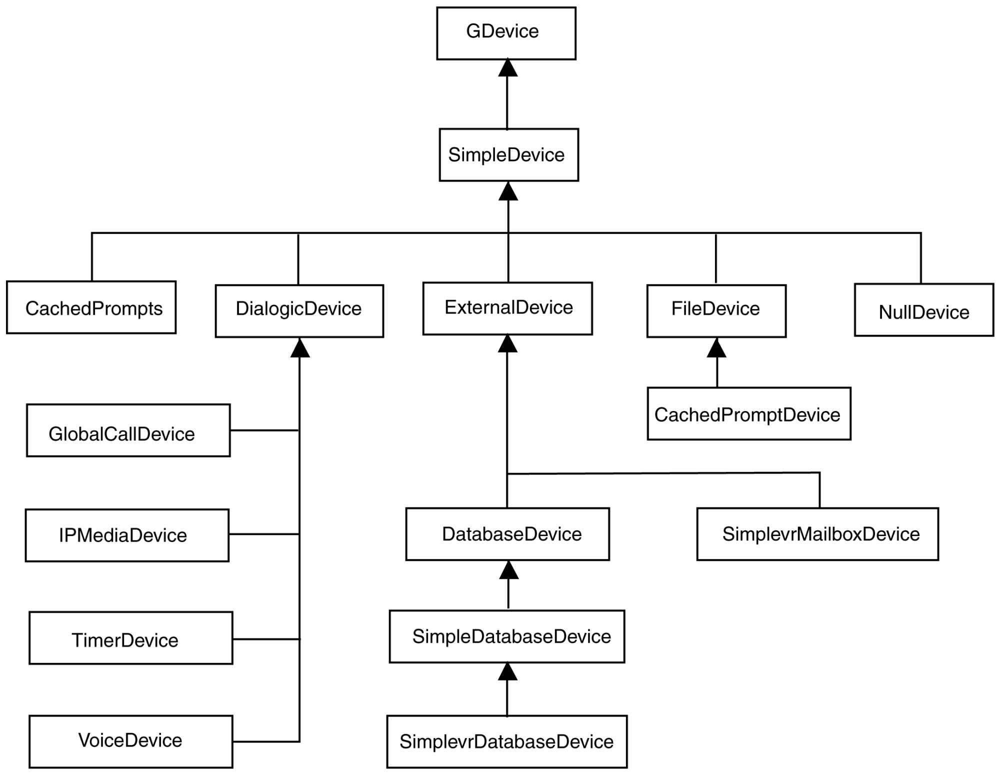
```
// Create an engine object
SimpleIvrEngine engine;
engine.create();
…

void SimpleIvrEngine::create()
{

        // Create all state machines
        for (int i = 0; i < NUM_CHANNELS; ++i) {
               SimpleIvrStateMachine* ivr = new SimpleIvrStateMachine;
               ivr.addDevices();
               addStateMachine(ivr);
        }
        // Create event waiter
        SimpleEventWaiter ew;
        setEventWaiter(&ew);
        // Kick the engine
        kick();
}

void SimpleIvrEngine::kick()
{
        m_ew->addObserver(this);

        // initial kick of event processor
        m_ew->run();

        m_ew->deleteObserver(this);
}
```

6.5 Simple Device Concept

Another important principle of SimpleIvr is the concept of SimpleDevice, which is a base for all derived entities that comprise the object space of the application. The entities are created either statically at the initialization phase or dynamically as needed. The majority of these entities represent physical devices from the Intel® Dialogic® domain, but some are pseudo-devices and are designed to provide external services (e.g. database, mailboxes, etc.).

A primitive SimpleDevice class itself implements an interface defined in an abstract class GDevice. One special device type, NullDevice, is introduced as a convenient device that does nothing, but is very helpful in application development to be returned instead of NULL pointer, which sometimes is inconvenient to check. Figure 24, “Simple Device Class Diagram” is a class diagram for all classes derived from SimpleDevice.

Figure 24. Simple Device Class Diagram



By having a lowest common denominator in the form of a SimpleDevice, these entities can be easily managed in a uniform fashion, and grouped together in so-called "alliances" to provide certain services together where one device is not capable of providing the service on its own. In other words, device alliance (SimpleDeviceAlliance class in the application) is a container which holds SimpleDevices and also bridges them betweeen each other.

For example, one may create a SimpleDeviceAlliance that will group a Global Call device, a database device (handle), and a cached prompt device to represent and serve as an intelligent object to a state machine instance. The concept of SimpleDevice and SimpleDeviceAlliance serves as a framework for application developers and is expandable to application-specific needs. The following code snippet illustrates the process of creating devices and alliances:

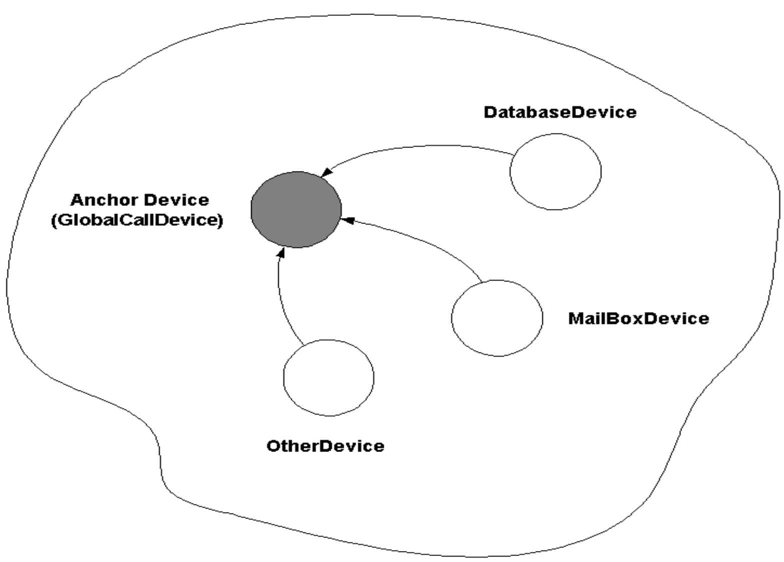
```
// Create all devices
GlobalCallDevice* gcd = new GlobalCallDevice;
VoiceDevice* vd = new VoiceDevice;
CachedPrompts* prompts = new CachedPrompts;
SimpleIvrDatabaseDevice db = new SimpleIvrDatabaseDevice;
SimpleIvrMailboxDevice* mailbox = new SimpleIvrMailboxDevice;

// Create device alliance
SimpleDeviceAlliance* alliance = new SimpleDeviceAlliance;

// Add all devices to the alliance, making GlobalCallDevice an anchor
alliance->addDevice(gcd, true);
```

```
alliance->addDevice(vd);
alliance->addDevice(prompts);
alliance->addDevice(db);
alliance->addDevice(mailbox);
```

It is important to mention one fundamental difference between devices from the Intel® Dialogic® domain and pseudo-devices. Devices from the Intel® Dialogic® domain are intrinsic SRL event-producing, whereas pseudo-devices cannot produce SRL events on their own. That could cause a problem when the pseudo-device needs to inform the state machine about some condition that would require the state machine's attention. The condition could be a normal completion event, e.g. asynchronous database retrieval request completed, or an error. It is very important to handle these types of events in a uniformed fashion with events from Intel® Dialogic® devices.

It is possible to directly place an arbitrary event to the SRL event queue. The only limitation is that the event posted to the queue must be a Intel® Dialogic® device. Second, the SimpleDeviceAlliance class, as designed in the way described above, allows devices in the alliance to discover each other. Therefore, a pseudo-device can delegate another device in the alliance (Intel® Dialogic® device) to post the event on its behalf to the SRL event queue. To facilitate this process, one Intel® Dialogic® device in each alliance is declared as an "anchor" device which carries event-posting payload in addition to its own functionality. Figure 25, "Simple Device Alliance Concept" illustrates the SimpleDeviceAlliance concept.

Figure 25. Simple Device Alliance Concept



To effectively operate, each state machine should be very reactive to the events they process; otherwise, it will hold up processing of the events by other machines and the system becomes non-reactive. State machines rely on SimpleDevices that are actually executing appropriate functions. Therefore, the following principle was laid as a foundation to the state machine implementation: Execution of actions by a state machine upon a SimpleDevice should not block the state machine execution of any other action upon another SimpleDevice.

R4 and Global Call APIs are designed with this philosophy in mind, providing two general ways of executing functions: synchronous and asynchronous. By calling these functions asynchronously an application satisfies the first rule. However, some APIs that the application utilizes may not be compliant with this rule. To overcome this obstacle, an application developer should come up with some technique to convert lengthy operations into a series of smaller steps. In some cases, applying this technique is not feasible due to the nature of the code, or simply because this is a third-party API.

In this case, a proxy programming pattern is handy. An application developer wraps lengthy functions into a piece of code that just initiates the function execution in a separate process or thread and immediately returns. Upon finishing operation, it notifies the state machine of its completion by posting an event to the SRL queue. Some operating systems have special facilities and functions as a part of their SDK that greatly help in implementation. They are called "thread pools". The following code fragment is provided as an example of using thread pools in Microsoft Windows* operating system with SimpleIvr application:

```
bool ASYNC_db_open(int devh, int* handlep, const char* name, int flags)
{
         bool bRet;
         // Aggregate parameters in a context structure
         DB_OPEN vContext = {
               devh,
               handlep,
               name,
               flags,
               0
         };
         // Create a new context and copy parameters
         PDB_OPEN pvContext = new DB_OPEN(vContext);

         // Keep track of outstanding worker thread requests
         InterlockedIncrement(&dwQueuedWorkItems);

         // Call Windows SDK to queue a worker item in the thread pool
         bRet = QueueUserWorkItem(
                       THREAD_ASYNC_db_open,           // worker item
                       pvContext,                      // function data
                       WT_EXECUTEINLONGTHREAD          // long wait
                       );
         return bRet;
}

DWORD WINAPI THREAD_ASYNC_db_open(PVOID pvContext)
{
         // Get the function parameters
         PDB_OPEN pContext = (PDB_OPEN)pvContext;
         long rc;
         int handle;

         // Call potentially lengthy function
         rc = db_open(pContext->name, pContext->flags);
         handle = rc;

         // Return parameters
         if (pContext->handlep != NULL) {
                       *pContext->handlep = handle;
```

```
        }
        // Post DB_EVENT_OPEN event to SRL queue
        DX_PUT_EVENT(pContext->devh, DB_EVENT_OPEN, 0, NULL, rc);
        // Get rid of context
        delete pContext;

        // Keep track of outstanding worker thread requests
        InterlockedDecrement(&dwQueuedWorkItems);

        return 0;
}
```

In the code above, a potentially lengthy operation, **db_open**, is called. It is represented by a proxy function, **ASYNC_db_open**, whose sole responsibility is to aggregate parameters into the call context structure **DB_OPEN**, and request operating system to queue **THREAD_ASYNC_db_open** to a worker thread in the thread pool. Then it immediately returns. The operating system schedules execution of **THREAD_ASYNC_db_open** function in the most efficient way. Upon completion of **db_open**, an event **DB_EVENT_OPEN** is posted to the SRL queue which then notifies an appropriate state machine of the completion.

6.6 Simple Device Properties

One of the goals of the SimpleIvr reference design application is to create it in such a way that it is configurable and adaptive to various device configurations and run-time scenarios. To satisfy this goal, a concept of properties was utilized and a corresponding abstract class, SimpleDeviceProperties, became a part of the application. SimpleDeviceProperties is a class that allows reading or writing properties to or from streams (files). All application entities may be configured by assigning a particular property class derived from SimpleDeviceProperties. It is illustrated in the following C++ code snippet:

```
// Global Call device properties
GlobalCallDeviceProperties gcdp;

// Open file with properties
ifstream is;
is.open("gc.props");

// Read properties
is >> gcdp;

// Create Global Call device
GlobalCallDevice gcd;

// Set device properties
gcd.setDeviceProperties(gcdp);
```

Devices are not the only entities that can be configured via properties. Other components, such as state machines, engines, etc., employ this concept of properties to achieve the desired flexibility.

Solution Configuration Testing and Performance 7

This chapter provides information on the following topics:

7.1 Solution Configuration Testing

The Intel® NetStructure™ Host Media Processing (HMP) software IVR solution was subjected to real-world test scenarios. To illustrate the system performance of the IVR reference (SimpleIVR) system, we have provided performance data regarding CPU utilization, heavy DTMF detection, and busy hour call attempts. This section will focus on the system described in Chapter 2, "Host Media Processing Configuration Description".

7.2 Test Methodology

The system was tested and key system level performance parameters were collected. Although many optimizations were utilized on this specific testing system, the individual applications and their system configurations may warrant further fine-tuning and thus, the material in this section should be treated as general guideline. You can also use the following data to help you choose the right platform that will provide the required system performance.

7.2.1 Performance Parameters

The following parameters are generally considered key benchmarks for an IVR system:

- Busy Hour Call Attempts (BHCA)
- Busy Hour Calls Completed (BHCC)
- Call lost rate
- Voice quality
- Digit detection errors
- Digit detection response time
- CPU utilization
- Memory utilization

Performance data was collected on the following parameters:

- Busy Hour Call Attempts (BHCA)
- Busy Hour Calls Completed (BHCC)
- Call lost rate
- Digit detection errors
- CPU utilization

7.3 Test Scenarios and Performance Data

Different test scenarios were used to show the full potential of the IVR system. These scenarios provide a view into the design envelope of the system in terms of density, performance and response time.

7.3.1 CPU Utilization

Performance information about CPU utilization includes the following:

- Hardware Platforms for CPU Utilization
- Test Scenario for CPU Utilization
- Performance Data for CPU Utilization

7.3.1.1 Hardware Platforms for CPU Utilization

The CPU utilization performance results are based on data taken from the hardware platforms defined in the following tables.

Computer 1

Parameter	Value
CPU	Intel® Pentium® 4 processor, 2.40 GHz
Model Information	P4N (Northwood) Pentium® 4 processor, 1.6 - 3GHz, 1.5 - 1.6V
Memory	512MB ECC RDRAM
Chipset	Intel® 850E Chipset
Level 2 Cache	512KB
Bus Speed (FSB)	4x133MHz (533MHz data rate)
NIC	Intel® 82801DB PRO/100 VE Network connection
OS	Microsoft Windows* 2000 Server SP3

Computer 2

Parameter	Value
CPU	Intel® Pentium® 4 processor, 2.40 GHz
Model Information	P4N (Northwood) Pentium® 4 processor, 1.6 - 3GHz, 1.5 - 1.6V

Parameter	Value
Memory	512MB DDR SDRAM
Chipset	Intel® 845G Chipset
Level 2 Cache	512KB
Bus Speed (FSB)	4x133MHz (533MHz data rate)
NIC	Intel® 82801DB PRO/100 VE Network connection
OS	Microsoft Windows* 2000 Server SP3

Computer 3

Parameter	Value
CPU	Intel® Xeon™ processor, 2.80 GHz
Model Information	P4N (Northwood) Pentium® 4 processor, 1.6 - 3GHz, 1.5 - 1.6V
Memory	512MB ECC RDRAM
Chipset	Dell* Computer Corp E7505 Chipset
Level 2 Cache	512KB
Bus Speed (FSB)	4x133MHz (533MHz data rate)
Memory Bus Speed	2x133MHz (266MHz data rate)
NIC	Intel® PRO/1000 MTW Network connection
OS	Microsoft Windows* 2000 Server SP3

7.3.1.2 Test Scenario for CPU Utilization

The following test scenario was used to collect CPU utilization performance data:

- Play: Two threads are simultaneously playing voice data from memory; each thread is responsible for half of the channels.
- Record: Two threads are simultaneously recording voice data to memory; each thread is responsible for half of the channels.

Configuration File

The following HMP configuration file was used for evaluating CPU utilization:

um_64-0-0.pcd

This file provides 64 voice channels.

Test Parameters

The test parameters included:

- Play format: 8 KHz/8bits per sample µLaw PCM voice played from memory
- Record format: 8 KHz/8bits per sample µLaw PCM voice recorded to memory
- RTP audio capabilities: G.711, µLaw, 64k, {20, 30ms} frame sizes
- AGC: On/Off

7.3.1.3 Performance Data for CPU Utilization

The following performance data regarding CPU utilization is grouped as follows:

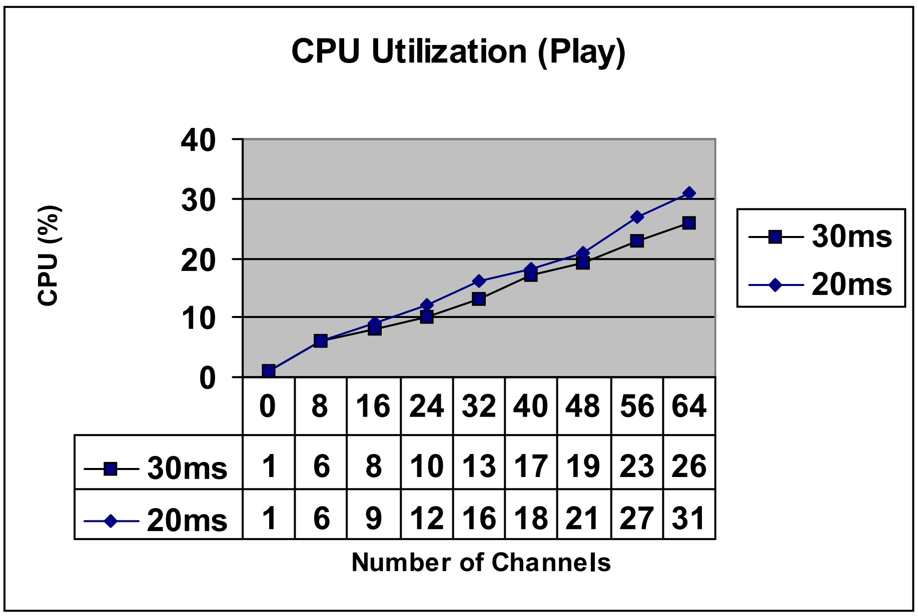
- Computer 1 Playing To/Recording From Computer 2
- Computer 2 Playing To/Recording From Computer 1
- Computer 3 Playing To/Recording From Computer 2

Computer 1 Playing To/Recording From Computer 2

The following graphs show the CPU utilization test results for:

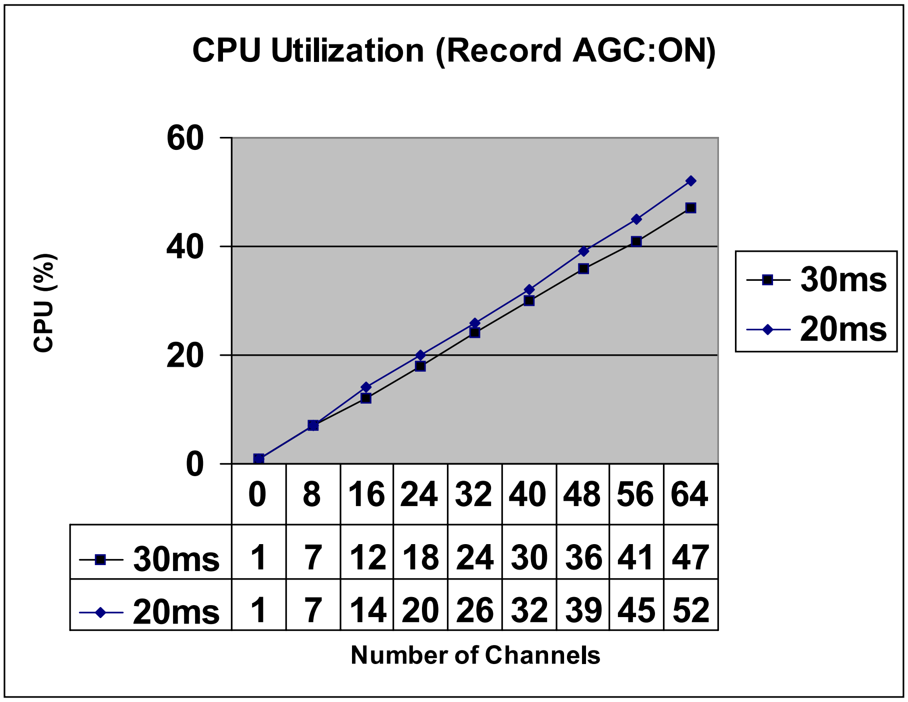
- Computer 1 playing to Computer 2 (Figure 26)
- Computer 2 recording from Computer 1 (AGC on) (Figure 27)
- Computer 2 recording from Computer 1 (AGC off) (Figure 28)

Figure 26. CPU Utilization (Play)



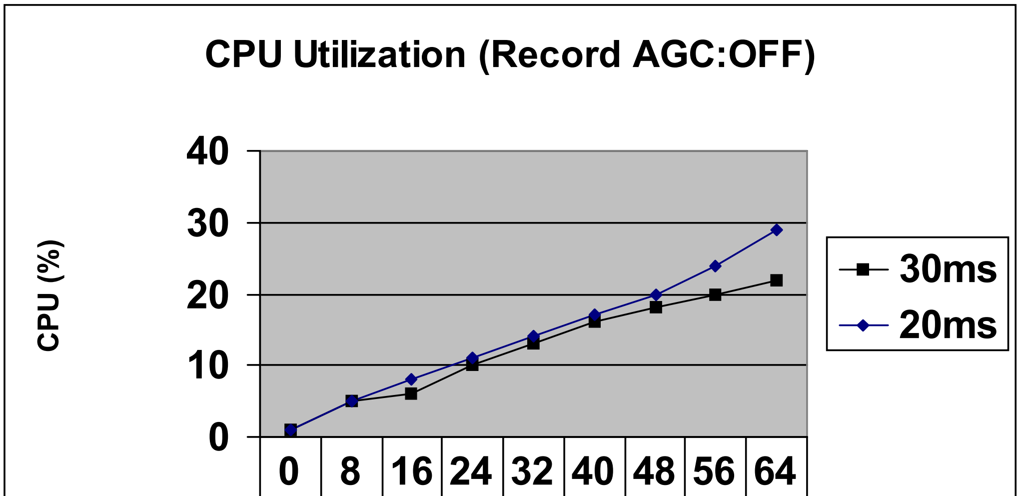
	0	8	16	24	32	40	48	56	64
30ms	1	6	8	10	13	17	19	23	26
20ms	1	6	9	12	16	18	21	27	31

Figure 27. CPU Utilization (Record with AGC On)



	0	8	16	24	32	40	48	56	64
30ms	1	7	12	18	24	30	36	41	47
20ms	1	7	14	20	26	32	39	45	52

Figure 28. CPU Utilization (Record with AGC Off)



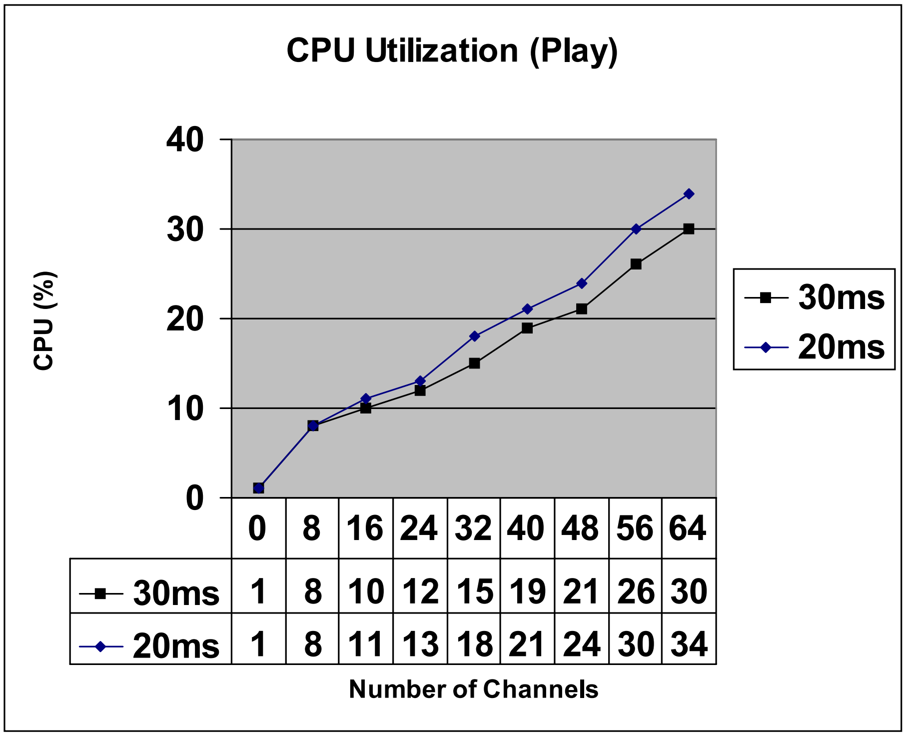
	0	8	16	24	32	40	48	56	64
30ms	1	5	6	10	13	16	18	20	22
20ms	1	5	8	11	14	17	20	24	29

Computer 2 Playing To/Recording From Computer 1

The following graphs show the CPU utilization test results for:

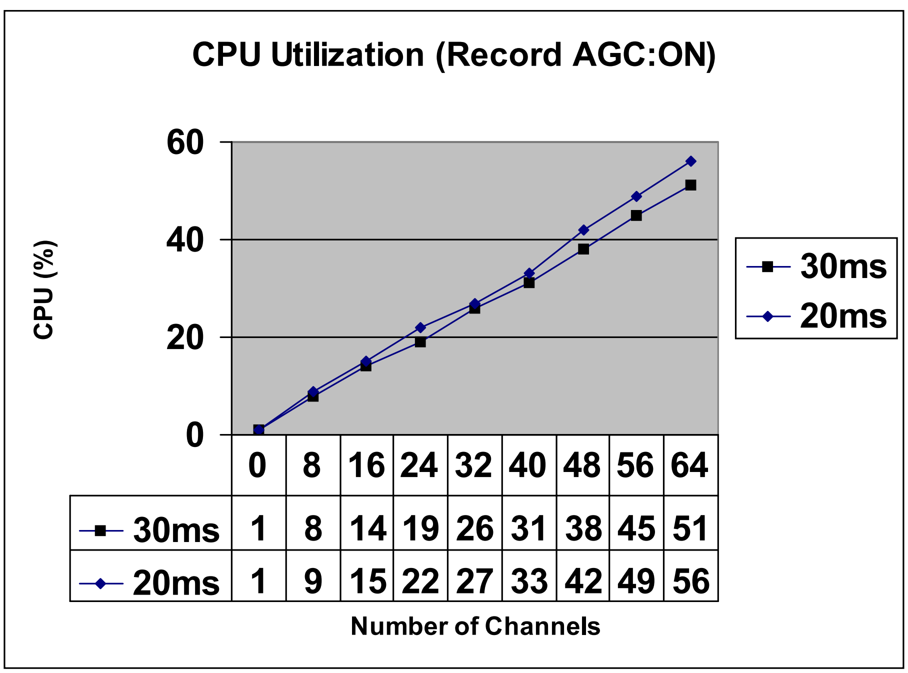
- Computer 2 playing to Computer 1 (Figure 29)
- Computer 1 recording from Computer 2 (AGC on) (Figure 30)
- Computer 1 recording from Computer 2 (AGC off) (Figure 31)

Figure 29. CPU Utilization (Play)



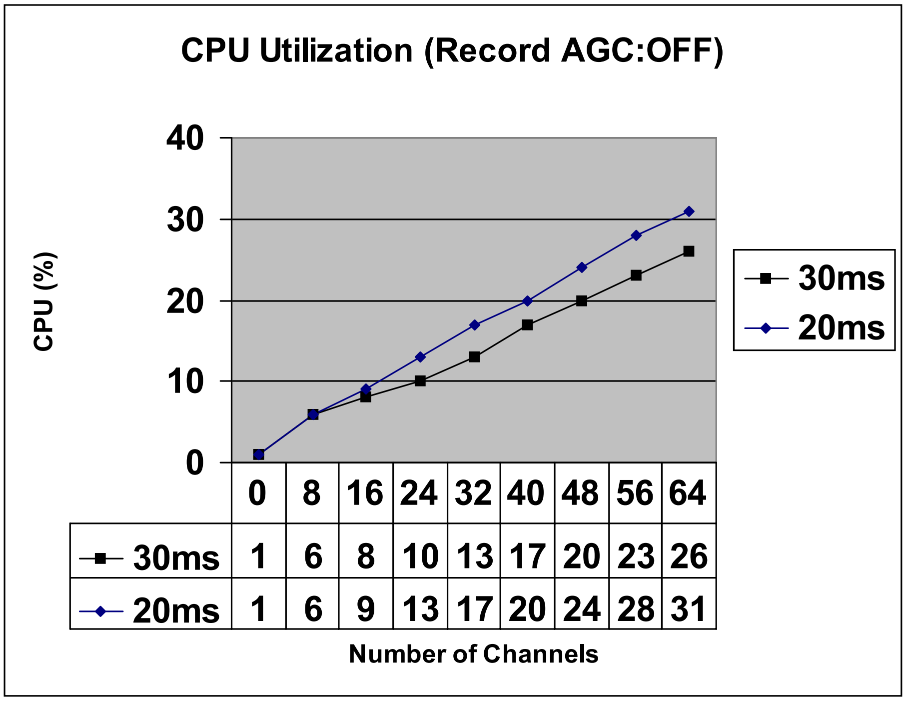
	0	8	16	24	32	40	48	56	64
30ms	1	8	10	12	15	19	21	26	30
20ms	1	8	11	13	18	21	24	30	34

Figure 30. CPU Utilization (Record with AGC On)



	0	8	16	24	32	40	48	56	64
30ms	1	8	14	19	26	31	38	45	51
20ms	1	9	15	22	27	33	42	49	56

Figure 31. CPU Utilization (Record with AGC Off)



Computer 3 Playing To/Recording From Computer 2

The following graphs show the CPU utilization test results for:

- Computer 3 playing to Computer 2 (Figure 32)
- Computer 3 recording from Computer 1 (AGC on) (Figure 33)

Figure 32. CPU Utilization (Play)

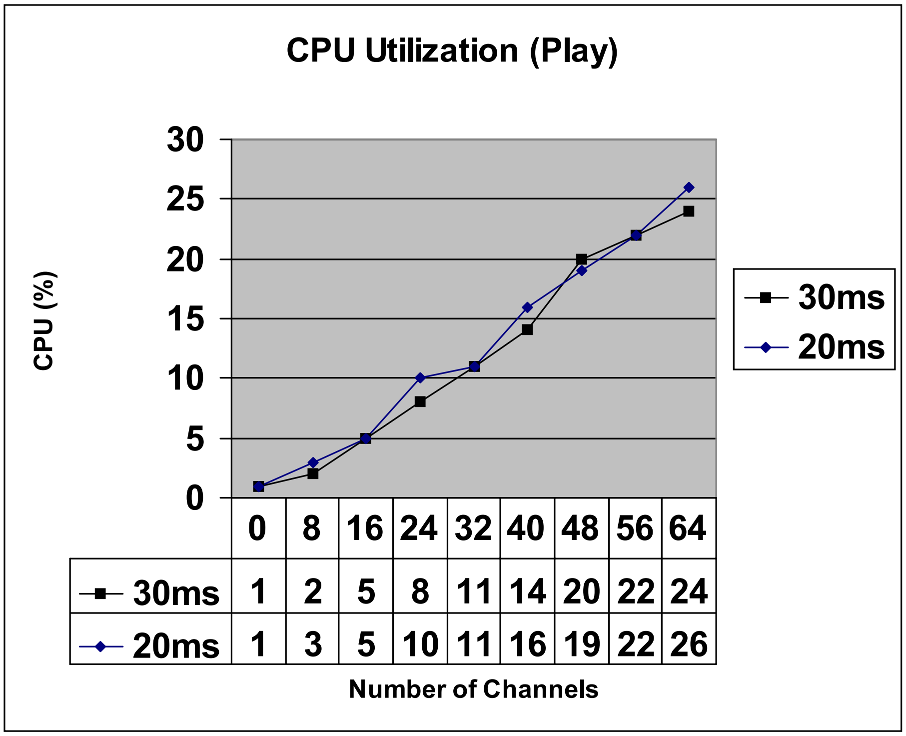
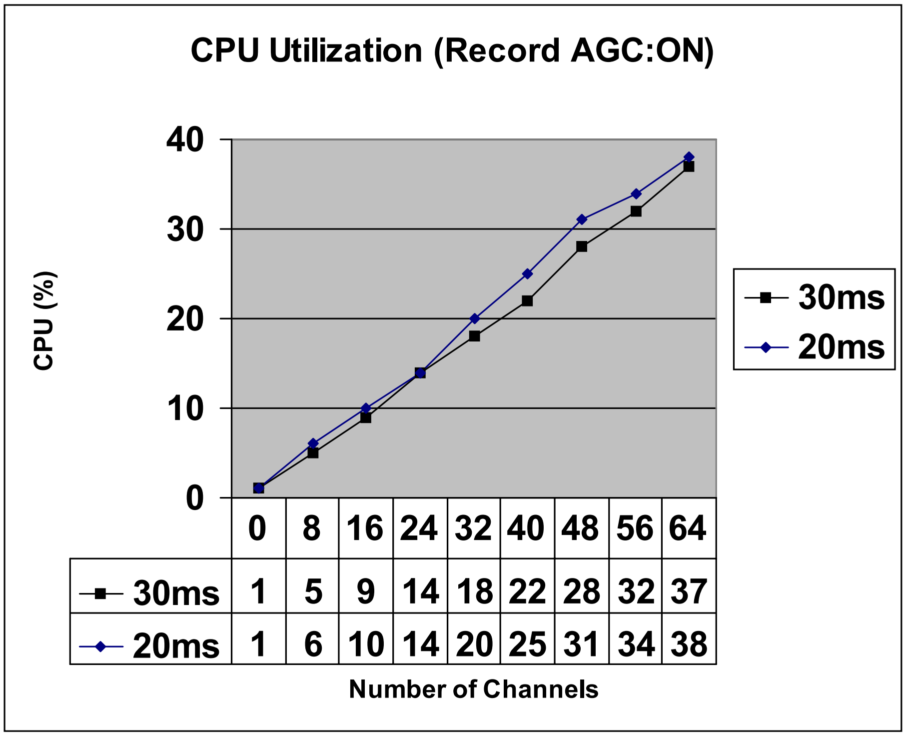


Figure 33. CPU Utilization (Record with AGC On)



7.3.2 DTMF Detection Test

DTMF detection performance is the most important parameter in an IVR system. Prompt detection and a low error rate is key to any IVR system. In this test, the computers were connected to a LAN on disparate network segments in a typical enterprise enviroment.

Performance information about DTMF dectection includes the following:

- Hardware Platforms for DTMF Detection Test
- Test Scenario for DTMF Detection Test
- Performance Data for DTMF Detection Test

7.3.2.1 Hardware Platforms for DTMF Detection Test

The following hardware platforms were used:

Computer 4

Parameter	Value
CPU	Intel® Pentium® III processor, 1.13 GHz
Model Information	P6T (Tualatin) Server Pentium® III-S Processor, 1.13 - 1.4GHz
Memory	512MB
OS	Microsoft Windows* 2000Professional

Computer 5

Parameter	Value
CPU	Intel® Celeron® processor, 850MHz
Model Information	P6C (Coppermine128) Celeron® Processor, 533 - 1.1GHz
Memory	256MB
OS	Microsoft Windows* 2000 Server

7.3.2.2 Test Scenario for DTMF Detection Test

Computer 4 and Computer 5 establish 64 full duplex, G.711 µLaw, 30ms RTP sessions using the IPML library. Computer 4 generates a series of random length strings of random DTMF digits and dials these digits using the INBAND mode. Computer 5 routes the digits back to Computer 4. Computer 4 detects the digits and compares them with the digits that were sent.

7.3.2.3 Performance Data for DTMF Detection Test

Over a period of 12 hours, with all 64 channels engaged, only 43 digit string comparison errors occurred out of 274,240 (<0.02% error rate). The errors were either due to dropped digits or digits that were detected twice.

7.3.3 Busy Hour Call Attempts (BHCA)

The main purpose of this test is to determine how many calls this IVR system can handle if:

- All calls come in at the same time
- All calls are very short (3 second hold time and 1 second intercall delay)

This extreme case has its real application such as call voting and the checking of instantaneous information.

Performance information about busy hour call attempts (BHCA) includes the following:

- Hardware Platforms for BHCA
- Test Scenarios for BHCA

- Performance Data for BHCA

7.3.3.1 Hardware Platforms for BHCA

Computers 1 and 2 as described in Section 7.3.1, "CPU Utilization", on page 74 were used for collecting performance data regarding BHCA.

7.3.3.2 Test Scenarios for BHCA

The systems (Computer 1 as the outbound system and Computer 2 as the inbound system) running the BCHA performance test application were able to execute H.323 calls at a rate of 48,750 calls/hour with 64 ports continuously executing Global Call cycles, with 3 second connection holding time and 1 second delay between calls. This equates to a rate of 762 successful call attempts per hour per port.

7.3.3.3 Performance Data for BHCA

The outbound system (Computer 1), running the BHCA performance test application on HMP, was able to successfully execute 853,000 calls in 63,000 seconds on all 64 ports with an error rate of 0.18% (1560 calls attempts failed). The inbound system (Computer 2), also running the BHCA performance test application on HMP, accepted all but the 1560 failed call attempts from the outbound system.

7.4 Configuring HMP with Intel® NetStructure™ PBX-IP Media Gateway

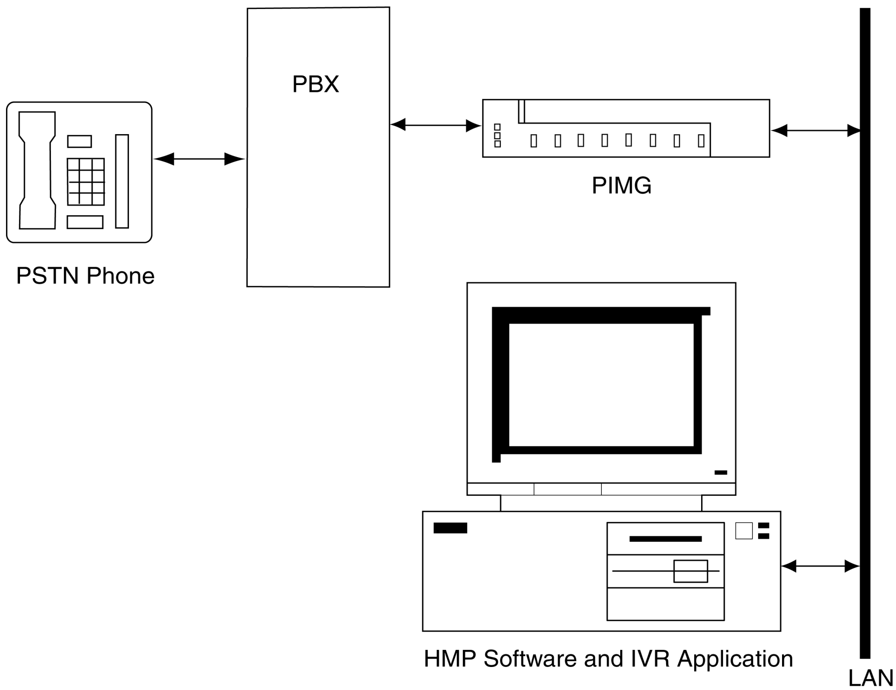
The HMP Software may be configured to interoperate with the Intel® NetStructure™ PBX-IP Media Gateway (PIMG) product. In this configuration, Figure 34, an analog phone connects to the PIMG through a PBX. The PIMG provides the gateway between the circuit switched network and the IP network. A computer connected to the IP network provides the IVR application. which, in this case, is the SimpleIVR application. This application uses H.323 as the call control protocol.

7.4.1 Configuration Considerations

In this configuration, certain PIMG parameters need to be set to specific values. The parameters and their values include:

- Operating Mode = H.323 Emulating
- Telephony Switch Type = none
- PCM Coding = μLaw
- Call Route Mode = Point-to-Point
- PBX Port/IP Endpoint Assignments = Assign appropriate IP terminal address for each port on the PIMG that is connected to the PBX or PBX emulator.
- Digit Relay Mode = Inband-Tone

Figure 34. HMP Configured to Interoperate with the PIMG



Application Design Alternatives for Your Solution 8

Intel® Telecom Services offers skilled resources to guide service providers through the solutions lifecycle – from planning and design through development and implementation. Intel consultants, experts in communications, provide an array of services, including network design, call and media flow analysis, application development, lab testing, and site implementation.

Intel consultants can help service providers evolve from their current telecom infrastructure to advanced modular network platforms, including the development of media servers and media gateways. Intel can help design and implement the network elements using Intel building blocks, including specifying and provisioning interfaces between networks and external devices and applications. With expertise in SS7, H.323, SIP, MGCP, Megacom, ISDN and TCP/IP technologies, Intel consultants can help service providers transition to the modular network.

Specific consulting services include:

- **Project Discovery Workshops** - Designed for developers in the early planning stages of a project, these workshops offer a comprehensive assessment of the project. The two- to three-day sessions help developers analyze technical requirements and provide guidelines for design specifications, potential architectures, operational considerations, implementation steps, and estimated resources.
- **Design Services** - During the design phase of a project, Intel consultants can develop design and functional specifications and/or test plans for a system or application.
- **Jumpstart Services** - Intel offers fast-paced, hands-on technology coaching, installation, and configuration guidelines, and a custom consultation to help solution providers integrate a wide range of Intel's hardware and software building blocks into their products or system designs.
- **Development Services** - Intel consultants can work with service provider development teams to incorporate a new feature or application into a solution. Deliverables typically include code, functional specifications, user guides, and associated consultation.
- **Lab Services** - Intel's laboratories provide state-of-the-art telephony environments for prototyping, testing, and fine-tuning applications with a variety of third-party switches and Intel equipment. Lab specialists develop and execute a series of test plans, including usability, performance, integration, load/stress, and feature conformance.
- **Site Implementation Services** - Site implementation services encompass a variety of remote and on-site installation and deployment tasks that require specialized network, speech and computer telephony integration (CTI) expertise. For example, Intel offers site installation services designed to help service providers install their SS7 platforms in the network provider's facility in a minimal amount of time. Intel consultants work with the service and network providers to correctly provision, configure and install the platforms within the network environment.

To learn more or to get started, send an email to ctcmarketing@intel.com or refer to the Consulting Services web page: *http://developer.intel.com/network/csp/products/4383web.htm*.

Index

I

K

L

M

N

O

P

R

S

T

U

V

To learn more, visit our site on the World Wide Web at **www.intel.com**

Intel Converged Communications, Inc.
1515 Route Ten
Parsippany, NJ 07054
Phone: 1-973-993-3000
Fax: 1-973-993-3093