The Dialogic® Diva® Software Development Kit (Diva SDK) offers a powerful set of Application Programming Interfaces (APIs) that can be used for implementing telephony applications, that is, applications that accept and process phone calls. The Diva SDK provides flexible tools for developers who choose to handle calls arriving via traditional telephony (PRI/E1/T1, BRI, and analog).

The Diva SDK is designed to work with the Dialogic® Diva® product family, which offers a full range of hardware products including T1/E1, ISDN-PRI, ISDN-BRI, and Analog.

The Diva SDK supports the various types of telephony through the same API, which means that applications can be written once and will then run using any physical media. This allows developers to implement an application that is connected to a mixture of T1/E1, BRI, and analog lines, for example. The application functionality can be the same for calls regardless of how they arrived at the application server. The operations available in the Diva SDK are provided as an abstraction above the Dialogic® Diva® Media Board, so that the lower layer software and hardware implement the actual details of the signaling protocol in use. This means that applications can be implemented without “knowledge” of the signaling protocols (Q-SIG, ETSI, SS7 and others).

Benefits Offered by the Diva SDK

The Diva SDK offers several benefits – high level APIs, which permit access to low-level features in case non-standard signaling is required, speech engine compatibility with other commercial speech engine products, and a choice of three different APIs to address specific criteria.

Addressing Real World Needs of a Communications Application

A notable strength of the Diva SDK lies in its design, whereby applications can be written using high-level interfaces; but if required, it allows use of function calls that give control over low-level signaling information. In this way, an application can support even an older PBX which does not comply with standards or one which implements proprietary features.

Speech Engine Compatibility

The Diva SDK has been used to interface a number of commercial speech engine products to Dialogic® Diva® applications. Any of the three Dialogic® Diva® APIs makes it possible to write one application and immediately have access to analog and digital telephony calls, all with the same executable file. The Diva SDK has a subsystem designed to make audio providers that can split call control from audio streaming, so that the audio goes to one thread, while the call control is performed by another thread.

This split means that a control subsystem (such as might be found in a VoiceXML/CCXML engine) can instruct the Diva API to make, break, and transfer calls, while the audio streams are taken directly from the Text-To-Speech (TTS) engine and sent to the Automatic Speech Recognition (ASR) engine. This can make the streaming of audio efficient even for large numbers of concurrent calls. Because customers may want to choose TTS and ASR from different vendors, the SDK supports two different audio providers for the voice streams (receive/transmit).

Choice of Three Different APIs to Address Various Requirements

The Diva SDK has three different APIs:

- Dialogic® Diva® API
- Dialogic® Diva® API for .NET
- Dialogic® Diva® Component API
Dialogic® Diva® API — This API is a high-level set of "C" functions for C/C++ applications. For many customers this is the most flexible of the APIs and allows the full range of Diva functionality to be used, including voice, speech, conferencing, unified messaging, fax, and data (modem). It is highly scalable and can be used for very large server applications, efficiently handling many concurrent calls.

Dialogic® Diva® API for .NET — This API provides a wrapper for all Diva API functions to allow access from managed code. It can be used from any .NET language, for example, C#, and contains an object oriented framework in C# as source code providing base classes for default handling. Various samples using this framework are provided with the Diva SDK.

The Diva API and Diva API for .NET are based on asynchronous function calls and events. This way of working offers developers powerful flexibility to arrange the system architecture as they choose. For example, it is possible to arrange for each concurrent call to be processed in its own thread or to handle the processing of calls in a single thread, being driven by API events. Event handling can be done by callback functions, by using a Windows event object or by using Linux signals, whichever is more convenient for the application architecture. When using callback functions, the source code of the application is independent from the operating system.

The Diva API has a full range of voice, speech, conferencing, unified messaging, fax, and modem functions, and can implement speech enhanced IVR, CTI, UM, or conferencing applications. The Diva API is also suited to implement Point Of Sales (POS) servers with low bit-rate modems or a Remote Access Server (RAS) with high-speed modems. The supported functions (for example, echo cancellation, automatic gain control, V90 modem) depend on the type of Dialogic® Diva® Media Board used. Highlights of the Diva API include:

- Initiating outgoing calls or processing inbound calls
- Processing of called and calling (DNIS, CLI), as well as redirected/redirecting numbers
- Control call ringing, reject, and line busy-out; answering calls based on timeslot/channel or called and calling numbers
- Call progress — Cause codes, ISDN call states, interpretation of in-band tones, fax status
- Tone detection — DTMF, busy, ring, SIT tones, modem/fax tones
- VAD — Onboard voice activity detection, for human speech and silence detection
- Answering Machine Detection
- Barge-in (DTMF or VAD trigger)
- Playing of prompts in A-law, µ-law, PCM, ADPCM, WAV or raw, including rewind, pause etc.
- Playing of continuous audio streams, for example, for music or TTS
- Control of volume and speed of playback
- Echo cancellation, for example, in ASR and conferencing applications
- Recording of audio to memory or disk (raw or WAV formats)
- Call transfer (tromboned, blind, supervised, ECT path replacement)
- Conferencing, including conference rights (whether members can talk or just listen), active talker detection and AGC mixing
- Supplementary services — Hold/Retrieve, transfer, CD, MWI, UUI, Facility, etc.
- User-to-User Signaling
- Sending and receiving facility messages at any time
- Full control over all bearer capabilities
- Efficient streaming of audio for ASR/TTS engines using "audio provider" API
- Send and receive fax (TIF, G3, color fax - JPEG), single and multiple fax files
- Concurrent handling of VoIP and traditional telephony calls, for example, E1, T1, Analog
- Detection and generation of generic tones under developer’s control
- Monitoring API, allowing passive call recording and logging of numbers
- Windows and Linux support — Allows source-compatible applications across platforms
- Microsoft SOA supported by third party “visual SDK”
In addition to voice and speech functions, the Diva API is also suitable for data applications, and supports protocols such as V.120 and modem (up to V.90) connections. For example, one of the code samples included in the Diva SDK download shows how to handle GSM SMS messaging with the Diva SDK.

Example 1 shows the code sample for C++ API playing audio prompts with barge-in. The code is short, well structured, and therefore easy to understand and suitable to be used as a boiler plate for similar applications.

```c++
// Sequence of a program to start a voice prompt. The application specific function
// "WaitForSignal" waits until the function "SendSignal" is called from the callback
// function called by the Diva SDK.
DivaSendVoiceFile(call->handle,"hello.wav",0);
WaitForSignal(call); //wait for prompt completion or barge-in
// Part of a callback function called "EventHandler". This callback function is
// has been registered with the Diva API via DivaRegister.
//
// void EventHandler(void *app, DivaEvent event, void *p1, void *p2)
// { CallInstance *call = (CallInstance*)p1;
//   //recover structure representing this call DWORD toneid;
//   switch(event)
//   { case DivaEventSendVoiceEnded:
//       SendSignal(call,PROMPT_DONE,0);
//       break;
//   case DivaEventDTMFReceived:
//       toneid = (DWORD)p2;
//       DivaStopSending(call->handle); //kill prompt play
//       SendSignal(call, BARGE_IN, toneid);
//       break;
//   case DivaEventToneDetected:
//       toneid = (DWORD)p2;
//       if(call->toneon && toneid==DivaEndOfTone)
//       call->toneon = FALSE;
//       else
//       { //process first non-silence event
//       DivaStopSending(call->handle);
//       //kill prompt play call->toneon = TRUE;
//       if(toneid==DivaHumanSpeech || toneid==DivaUnknownTone)
//       SendSignal (call,BARGE_IN, SPEECH );
//       else if(toneid==DivaEndOfTone)
//       SendSignal (call,BARGE_IN, toneid);
//       } break;
//   } } Example 1. C/C++ API Playing Audio Prompts with Barge-in Code Sample
```

**Dialogic® Diva® Component API** — This is an ActiveX (or COM) API for direct integration in .NET or VB.NET applications (VB 6 is supported as well). The Diva Component API can be used with a wide variety of ActiveX-based development tools (in addition to the VB family) including Borland Delphi. It supports synchronous processing, which can make development easier. The Diva Component API is often used for fast implementation and “proof-of-concept” prototyping. The Component API does not have restrictions in terms of functionality or in performance, as critical tasks are handled in the API. Therefore, it is well suited for production systems.
The DivaCall object of the Diva Component API offers basic functionality for speech applications, including outgoing and incoming calls, playing prompts, send/get DTMF, recording to disk, call transfers and fax send/receive. Applications can be programmed using asynchronous or synchronous call models and can be event-driven or polling, depending on the requirement. The Diva Component API is well suited for prototyping IVR systems, even those including advanced features such as echo cancellation and barge-in. The Diva Component API also offers a DivaSystem object that supports multiple call instances per application, which in turn allows for building an efficient multi-call server application in VB.NET.

Example 2 shows the code sample of a simple answering machine application in VB.NET. Being a VB.NET application provides that even developers who are not “communication application” experts can develop, maintain, and enhance such programs.

```
Sub MyCall_OnIncomingCall() Handles MyCall.OnIncomingCall
    Dim retVal As DivaResultCodes
    Dim Tone As Object
    Dim Repeat As Long
    Dim RecordFile As String
    If (bShutdown = False) Then
        bActive = True
        theParent.LogChannel(logChannel, "Answer incoming call")
        retVal = MyCall.Answer()
        If (retVal = DivaResultSuccess) Then
            theParent.LogChannel(logChannel, "Connected stream announcement")
            retVal = MyCall.SendVoiceFile("DefaultGreeting.wav")
            If (retVal = DivaResultToneDetected) Then
                RecordFile = "Msg _ " & MyCall.CallingNumber & " _ " & logChannel & ".wav"
                theParent.LogChannel(logChannel, "start recording to " & RecordFile)
                retVal = MyCall.RecordVoiceFile(RecordFile, DivaAudioDefault, 20, 0, "")
                theParent.LogChannel(logChannel, "recording ended with reason " & retVal)
            End If
            Call MyCall.Disconnect()
        End If
    End If
    bActive = False
End Sub
```

Example 2. A Simple Answering Machine Application in VB.NET

**Note:** The Extended CAPI API is offered by Dialogic. It is compliant with the CAPI 2.0 standard and therefore not considered to be a Diva SDK API. The Extended CAPI API offers Dialogic-proprietary extensions to CAPI 2.0 for handling supplementary services, call progress, call tromboning, modem handling, and other useful features. This API allows widespread cross-vendor compatibility, and can also support legacy CAPI applications. Furthermore, existing CAPI 2.0 applications can be enhanced by using the Extended CAPI API.

**Diva SDK Materials**

The Diva SDK contains materials needed for those choosing to develop voice, speech, fax, conferencing, or other voice/data applications.

**Diva API and Diva API for .NET**

The Diva API and Diva API for .NET have the following materials:

- Header files and libraries for “C” or C++ (DLL and Linux .o)
- Documentation in PDF and Windows HTML Help format
- Wide range of samples covering telephony, conferencing fax, and data. Basic samples are available (for console mode testing), as are fully working GUI samples with separate GUI and call processing modules.
- Samples cover Win32 and Linux platforms
The Diva SDK (Windows version) is targeted on Microsoft Visual Studio .NET, but has been used successfully in many development environments.

Diva Component API

The Diva Component API has the following materials:

- Component DLL providing Diva objects to import into the development system
- PDF and Windows HTML Help documentation describing available objects and methods
- Sample programs for VB, VB.NET and VB script
- Samples cover fax, voice and data, and feature single and multiple call examples

Other Materials

The Diva SDK package contains documentation about the CAPI 2.0 API. It also contains documentation for the Diva Management Interface, which is an API that allows applications to set parameters and retrieve information from Dialogic® Diva® Media Boards including detailed configuration information, hardware status, and call statistics.

Obtaining Diva SDK

The Diva SDK is free-of-charge and can be used by owners of Diva Media Boards. The Diva SDK can be downloaded from the Dialogic website at http://www.dialogic.com/en/products/media/development-tools/diva-sdk.aspx.