
Selecting the Right Tool

**Key considerations for
Computer Telephony and Speech
Application developers**

Introduction

Even in the current economic climate, the market for Computer Telephony (CT) and speech applications is growing, as organizations are looking for ways to lower costs, improve customer service and optimize operational efficiency. Speech applications, in particular, are finding increased market acceptance. After several years of technological progress and decreasing cost, TTS (Text-to-Speech) and ASR (Advanced Speech Recognition) technologies have reached a point where their widespread deployment becomes increasingly feasible. Together, they are enabling new application types and make existing telephony applications even more user friendly. And although 'traditional' DTMF (i.e. touchtone) applications are often overlooked, they too represent a significant and growing market, as many solutions can effectively be implemented without the expense of speech. In total, the Voice Business marketplace is forecast to grow from an estimated US \$632m in 2001 to \$4.3bn in 2007, representing a healthy compound annual growth rate of 38% (Source: Datamonitor). For developers, this opens a wide range of attractive, high value opportunities, such as IVRs, call center solutions, outbound dialing and survey applications, voice portals and others.

Developers looking to enter the voice business market face a number of hardware and software choices. Chief among these decisions is the choice of development environment – it can determine success or failure of a development initiative and make the difference between a profitable project rollout and a costly failure. This document highlights some of the critical issues that should be considered.

A summary of the criteria and the relative positioning of VBVoice relative to these can be found in the appendix.

Programming Environment

The ideal programming environment should accomplish a number of goals: it should maximize developer productivity and freedom of application design while offering a rich set of debugging capabilities and ease of use. To maximize ease of use, most telephony and speech toolkits offer a degree of visual programming through a 'drag-and-drop' style interface.

However, while visual programming has the potential to greatly enhance *Ease of Use*, it can also become a limiting factor. A *short learning curve* sometimes comes at the expense of other developer productivity aspects that may only become obvious after the tool is in use. *Extensibility* is chief among these. The developer's ability to extend the visual programming environment to incorporate other components not provided by the toolkit or to take full advantage of new hardware and software is critical for most real-world telephony applications. For example, the ability to integrate with .NET environments and applications is an increasingly significant decision criterion.

As a result, careful consideration should be given to the ability to integrate third party products, to add custom programming and to further customize the standard controls provided by the tool. To further shorten the learning curve for developers new to telephony and speech, the programming language should ideally be industry-standard (such as C#, VB.NET, Java) and not proprietary to the selected tool. While most development environments for computer telephony applications allow for the addition of external functionality, some require knowledge of proprietary scripting languages or handling of complex, low level programming. Others enable developers to leverage their programming knowledge and expand on the telephony controls included in the toolkit through custom code, as well as make use of third party components, such as ActiveX.

Finally, the strength of the debugging environment, and the ability of source level debugging is of critical importance. Attempting to uncover call flow or recognition problems in an application without a rich set of tools sharply reduces developer productivity and increases error rates of the final application. Ideally, the chosen programming environment should tie into an industry-standard debugging platform that you or your developers are already familiar with. A further issue associated with debugging should be the tool's ability to generate sophisticated call log files for further analysis.

Together these characteristics allow developers to create sophisticated and powerful applications without a long learning curve and ongoing trial-and-error during application development.

How VBVoice Measures Up:

Serious applications require serious development tools. Therefore, VBVoice has been designed from the ground up for Microsoft Visual Studio and Visual Studio.NET, the industry-standard development environment for Windows platforms. VBVoice is fully integrated with its sophisticated features such as IntelliSense(TM), a powerful debugger with real time logging, and full ActiveX support. Our drag-and-drop components can be fully customized and extended through custom code, written directly in Visual Basic, C# or the other languages supported in Visual Studio.NET. And with millions of Visual Basic programmers worldwide, creating VBVoice applications requires a minimum of training.

At the same time, VBVoice offers more than just powerful languages. Out of the box, VBVoice comes with a complete, fully event-driven program framework, specifically designed for call processing on multiple channels. The framework relieves the programmer from making such low-level design decisions as multi-threading model, synchronized access to variables, inter-channel communications, etc. In addition, VBVoice provides a structure to organize the source by attaching the code directly to the graphical controls. This allows identifying the relevant sections of logic simply by double clicking on the control's icon.

As the Windows environment is extremely rich and continuously evolving, this seamless integration of VBVoice with Visual Studio gives you the best of both worlds: a comprehensive and flexible Computer Telephony and speech toolkit that is supported by arguably the most modern and powerful general purpose development environment for the Windows platform.

VBVoice call-flows are compiled directly into the machine's native binary code, which runs (as any other Windows executable) at a full CPU speed. There is no need for any form of run-time interpretation. Multithreading is implemented in the VBVoice runtime engine. The well-documented model lets developers apply serialized or fully multithreaded modes to the whole application or to individual controls.

Architecture

Most telephony and speech applications are business-critical in nature. They serve as auto-attendants, automate routine processes in call center environments, schedule employees and much more. Failure of an application to scale to required levels and to survive hardware problems is not an option. The architecture of your platform should offer proven scalability and the ability to hot-swap applications. It is important that applications can execute independently from each other and from the system processes, can be hot-swapped, easily provisioned and configured. In hosting environments or in situations where multiple applications are being deployed, the platform should enable the sharing of telephony resources (i.e. speech licenses and telephony hardware) across multiple independent applications. Such a distributed

architecture also allows individual applications to be interrupted for upgrades or other maintenance without interrupting other applications on the same server.

While some environments allow the sharing of telephony hardware and hot-swapping of applications, developers should ensure that the platform does not create a monolithic executable for multiple applications. In such a system, the failure of one module could stop the entire system, as the monolithic executable is only as reliable as its weakest module. While your code may be bulletproof, can you guarantee the same for all the components and libraries you have to use? Even under normal conditions, such an architecture causes problems: a database call by one module essentially puts all other modules on hold.

How VBVoice Measures Up

VBVoice 5.0 provides support for a fully distributed Master-Slave architecture. The system resources and services (telephony hardware, TTS and ASR engines, etc.) are hosted on one or more Master machines. The applications (Slaves) may run not only in separate processes, but also on separate machines. If necessary, a Slave machine may be 100% dedicated to a single application and run with no local system resources whatsoever. Still, the underlying communication across the network is completely transparent to the application logic. This provides for ultimate separation, load balancing, easy hot-swaps, independent provisioning and scalability.

Telephony

The world of telephony and speech applications is a complex one, with multiple hardware and programming interfaces and a plethora of standards. The resulting learning curve for new developers tends to be very steep. This is where rapid application development tools really shine. Their controls encapsulate and abstract common call processes to simplify application development and shield the developer from hardware specific programming. In evaluating your application development tool, you should look not only at the raw number of these controls, but rather at their depth, customizability and extensibility. Otherwise, you may find that the feature you are looking for is simply not doable in the environment of your choice. Also, the tool should provide support for multiple hardware vendors, advanced protocols such as ISDN and VoIP and offer a comprehensive lineup of call control features for your particular application. For example, call queuing, agent monitoring, recording and conferencing capabilities are significant in call center applications, while other solutions may require broad fax support, web integration, switch integration via TAPI, etc.

How VBVoice Measures Up

VBVoice is focused on voice applications. It offers a number of controls that go beyond basic call processing, many of them designed specifically for call-centers: CallQueue (a host-based ACD), TapRecord (for agent monitoring and call center quality control), CallTimer (for flexible call timing and billing), AgentX and Remote Control (for agent screen pops), Voice Commands (for hot-word triggered global menus in speech recognition) and more. VBVoice also supports TAPI and Wave Driver standards, now the preferred way of integration with the latest generation of IP-PBXes.

Speech Processing

The market for speech recognition engines and Text-to-Speech engines is continually evolving. Consequently, the development platform should support multiple TTS and ASR engines, letting you select the appropriate engine for each individual application development effort. Key here is again the degree of support for native interfaces to allow for maximum control. For example,

relying solely on SAPI as the interface to multiple TTS engines may result in problematic implementation. This is because the generic SAPI interface works differently with different engines. Fine variations in timing, buffering schemes and performance, for example, can result in irritating gaps, clicks and delays.

Additional speech capabilities to look out for: does the tool support speaker verification capabilities of one or more speech engine vendors, and what other areas of speech deployment are being supported? Dynamic grammars, for example, can be used to simplify complex application development.

Consideration should also be given to the tool's ability to record and to dynamically alter voice prompts for your applications. This is particularly important for personalized call processing, depending on caller preferences, payment options etc. In these circumstances, prompts and call-flows should be easily manipulated at runtime. Needless to say, the recording and editing of prompts should be an integrated capability.

How VBVoice Measures Up

When it comes to speech processing engines, the "one size fits all" approach is not good enough. This is true for ASR, but even more so for TTS. Therefore, we've spent a lot of effort to carefully research and evaluate different vendors and their interfaces before adding their product to VBVoice.

*For **speech recognition**, we have partnered with two undisputed industry leaders: Nuance and SpeechWorks. VBVoice 5.0 allows access to the full power of the engines: Voice Recognition, Speaker Verification, Dynamic Grammars, Voice Commands (for global, "always-on" commands) and Natural Language Processing.*

*For **Text-to-Speech** capabilities, it is our experience that superior results require individually crafted, native API interfaces (although VBVoice also offers a SAPI interface). Consequently, VBVoice supports a range of TTS engines that have passed stringent evaluation and testing criteria. They include:*

- 1. Fonix 5.1 (English, German, Dutch, French, Italian and Spanish) as the main multilingual solution.*
- 2. Speechify, offering distributed architecture and excellent quality (English, Spanish, and soon in 2.1 French, German and Japanese), for high call volume installations.*
- 3. L&H TTS 3000 for outstanding Japanese support*
- 4. L&H RealSpeak Telecom for a unique combination of quality, performance (distributed architecture) and good language coverage.*
- 5. Lucent for price sensitive applications.*
- 6. In combination with Aculab hardware, support for the free Aculab TTS is the newest option added to the VBVoice environment*

*Although we provide a full-featured **voice-editing** tool (Announce!), in VBVoice, prompts may be recorded and edited directly from within the Visual Studio IDE. Furthermore, prompts can be dynamically changed at run-time, by the Visual Basic or C# code of your application.*

Deployment Considerations

Another important consideration for developers is the ease of product licensing and deployment. This is particularly true for commercial application developers, such as system integrators and

ISVs. Ideally, deployment of a finished application and a later increasing in scope should be as painless as possible, yet assure the developer of the licensing integrity of the finished product. Consider whether the platform requires the use of “Dongles” for commercial application deployment, or whether the process is software-enabled. The latter approach allows for easy upgradeability and also has the potential for the re-licensing of developed applications to other customers.

How VBVoice Measures Up:

In Version 5, VBVoice introduced a sophisticated software module, the Runtime Manager to handle all aspects of software licensing and deployment. No hardware dongles are needed and to increase the size of a licensed application or to add new capabilities such as TTS or speech recognition requires a simple software upgrade. A forthcoming release will also support the re-licensing of custom application code to additional customers, enabling our application developer customers to further safeguard their intellectual property.

Support for Standards

Depending on your organization, standards such as VoiceXML and the forthcoming Microsoft-SALT can also play an important role in your tool selection. In the author’s opinion, however, other considerations as outlined in this document will likely be more important in the day-to-day development work. In addition, standards may impose restrictions on your development effort, since certain new developments and/or specific capabilities that you are looking for have not yet been reflected in a generally slower evolving public standard. VoiceXML, for example, has an acknowledged weakness in the area of call control, a fact that only recently has been addressed with the issuance of CCXML (Call Control XML) by the W3C. For the foreseeable future, other platforms will continue to exist, and some of them are beginning to extend to support standards environments.

How VBVoice Measures Up:

VBVoice currently does not support VXML or SALT environments. We believe that, for most developers, the return on investment of a development effort and the “time-to-money” question are generally more significant considerations. However, we also recognize that standards will continue to play an increasing role in the future of telephony and speech applications and future versions of VBVoice will offer developers a way to extend applications into the world of standards if and when required.

Conclusion

As you can see from this short document, the choice of a development tool for your telephony and speech application is subject to a number of criteria, each of which requires careful consideration. The ideal toolkit should offer a satisfactory answer in all of the outlined categories and excel in the ones that *count* in your particular scenario.

For almost 10 years, Pronexus has continued to deliver the most comprehensive and flexible rapid application development tool for telephony and speech applications: VBVoice. We are dedicated to its continued evolution to keep it at the forefront of a changing market. Our developers’ innovation and attention to customer requirements is what sets VBVoice apart.

APPENDIX

Evaluation Criteria	VBVoice	Company B	Company C
Programming Environment			
<i>Industry-standard programming language</i>	Programming in VB, VB.NET, C#, eliminates the need to learn proprietary languages and shortens the learning curve.		
<i>Source level debugging</i>	Integration with Visual Studio allows line-by-line execution, breakpoints, debug screens, etc., giving developers thorough control over application debugging		
<i>High level programming</i>	Complete, fully event-driven framework, relieving programmers from low-level design decisions and time-consuming implementation		
<i>Extensibility (interface to 3rd party products and technologies)</i>	Effortless integration with existing applications and libraries using Web Services, COM+, ActiveX controls and custom DLLs.		
<i>Customizable voice controls</i>	Functionality of all controls can be customized by adding VB, VB.NET or C# code to the events.		
<i>Integration of next generation technologies</i>	Fully integrated into Visual Studio .NET, the latest Integrated Development Environment (IDE) from Microsoft.		
<i>Reusable custom modules</i>	Taking advantage of the Visual Basic open architecture; allows the creation and import of VBVoice Composite Controls (COM objects) that can be shared by various applications.		
Architecture			
<i>Modularity/distributed architecture</i>	Break applications into multiple cooperating modules, distributed over a LAN for increased programming flexibility and redundancy.		
<i>Sharing of telephony resources among several applications</i>	System resources are hosted on one or more master machines while the applications themselves run in separate processes on separate machines. This allows sharing of telephony hardware and TTS and ASR engines among a large number of applications, resulting in reduced system costs.		
<i>Scalability</i>	Modularity feature: build and run as many applications as necessary. Add processing power to the application by adding PCs to the LAN		
<i>Independent application provisioning and execution</i>	Development, provisioning and execution of an application are the same as in stand-alone mode; Individual applications can be terminated or hot-swapped without affecting the rest of the system		
<i>Multithreading</i>	Make calls to slow databases or mainframes without affecting user experience on other channels.		
Telephony			
<i>Multi-vendor telephony hardware support</i>	Choose from a variety of telephony hardware vendors including Aculab, Dialogic, and Brooktrout.		
<i>PBX integration</i>	PBX integration allows to build call-center applications taking advantage of advanced features of the switch		
<i>TAPI support</i>	Supports TAPI, the preferred interface to the latest generation of IP-PBX switches. With many modern PBXs, this eliminates the need to purchase proprietary voice cards.		
<i>Voice over IP</i>	Seamless support for voice calls over H.323 using Dialogic card and Aculab VoIP. Save on long distance charges.		
<i>Faxing capabilities</i>	Included in VBVoice, VBFax provides necessary tools to easily add faxing capabilities to an application.		

<i>Advanced protocols</i>	Through partnerships with Dialogic and Aculab, VBVoice supports a wide range of advanced protocols, including T1 ISDN, Euro ISDN, R2-(CAS), DPNSS, and QSIG. Other external features such as single line digital transfer or TBCT, are also supported.		
<i>Call queuing</i>	CallQueue control provides call distribution (ACD) functionality, such as call routing, , transfers, and messaging on-hold. Useful for call center applications.		
<i>Agent monitoring</i>	Monitor quality of Agent's call handling via the TapRecord control.		
<i>Conferencing</i>	Allows multiple voice card lines to be merged into an audio conference, presented in an easy to use Conference Control.		
<i>Web query</i>	WebQuery control allows search for data within a web site matching a search criterion.		
<i>Multilingual support</i>	Language Control supports Multiple languages with out any programming		
Speech Processing			
<i>Integration with leading speech recognition engines</i>	Integrates with leading speech recognition engines from Nuance and Speechworks (including SpeechWorks OSR).		
<i>Dynamic grammars</i>	Grammars can be modified on the fly for a running application, eliminating the need to shut down for system tuning and modifications.		
<i>Integrated sound editing studio</i>	Announce!, a full-featured sound-editing studio for the recording of high quality prompts. Also, record prompts from within Visual Studio. Prompts can be dynamically changed at runtime, allowing personalized call processing		
<i>Speaker verification</i>	Optional Verify Speaker control validates a caller's identity on the basis of his/her unique speech characteristics, resulting in unsurpassed security and convenience for users.		
<i>Voice commands</i>	An optional control in VBVoice, VoiceCommands allows system-wide voice commands (such as 'Help, Operator, Back, etc.) for the duration of a call.		
<i>Integration with multi-vendor text-to-speech (TTS) engines</i>	Choose from a variety of TTS engines including Fonix, Speechify, L&H, and Lucent.		
Open Database Connectivity			
<i>Open database connectivity</i>	Leverage existing database infrastructure through compatibility with any ODBC data source.		
<i>WAP application development</i>	Extend development capabilities beyond traditional telephony applications, giving you the option to develop applications for WAP enabled devices, such as PDAs, Blackberries, and cell phones.		
<i>Remote system monitoring</i>	Networks package allows remote control of telephony server and real time monitoring of line status, call statistics and system performance over TCIP/IP and the Internet.		
<i>Client/server application communication</i>	Exchange data between applications on a network. Commonly used in call center applications to deliver information to an agent's desktop via screen pops.		